



NRL/MR/8120--02-8613

Antenna Design, Modeling, and Testing on the WindSat Satellite Wind Direction Measurement System

WENDY LIPPINCOTT

*Advanced Systems Technology Branch
Space Systems Development Department*

TED GUTWEIN

HOMER BARTLETT

MIKE SMYTHERS

*Microstar Inc.
Melbourne, FL*

PETER GAISER

*Remote Sensing Physics Branch
Remote Sensing Division*

BILL PURDY

*Purdy Engineering
Poolesville, MD*

March 29, 2002

20020510 085

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE March 29, 2002	3. REPORT TYPE AND DATES COVERED		
4. TITLE AND SUBTITLE Antenna Design, Modeling, and Testing on the WindSat Satellite Wind Direction Measurement System		5. FUNDING NUMBERS 81-B246-E1-5		
6. AUTHOR(S) Wendy Lippincott, Ted Gutwein,* Peter Gaiser, Homer Bartlett,* Mike Smythers,* and Bill Purdy†				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Research Laboratory 4555 Overlook Avenue, SW Washington, DC 20375-5320		8. PERFORMING ORGANIZATION REPORT NUMBER NRL/MR/8120--02-8613		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSORING/MONITORING AGENCY REPORT NUMBER		
11. SUPPLEMENTARY NOTES *Microstar Inc., Melbourne, FL 32934 †Purdy Engineering, Poolesville, MD				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		12b. DISTRIBUTION CODE		
13. ABSTRACT (Maximum 200 words) WindSat is a remote sensing spacecraft designed to be a spaceborne demonstration of passive microwave polarimetry to measure ocean surface wind speed and direction. Polarimetric radiometry measures the Stokes vector, which provides information needed to retrieve the ocean wind vector. The design, modeling, and measurement of the reflector antenna system used to perform the radiometry will be discussed in this report.				
14. SUBJECT TERMS Antenna Wind direction measurement Remote sensing		Satellite Stokes' vector Microwave polarimetry		15. NUMBER OF PAGES 100
				16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

TABLE OF CONTENTS

SECTION	PAGE
I. INTRODUCTION	1
II. WINDSAT REFLECTOR DESIGN	4
III. HORN DESIGN AND MEASUREMENT	14
IV. RANGE MEASUREMENTS	23
V. COLD AND WARM LOADS	41
VI. COMPUTER CODES FOR STOKES VECTOR CALIBRATION	44
VII. SUMMARY	49
ACKNOWLEDGMENTS	49
REFERENCES	50
Appendix A. Radiometer Performance Overview	51
Appendix B. Reflector Modeling Codes	56
Appendix C. Comparison of OSUREF and GRASP8w Codes	58
Appendix D. POE Code for +/- 45 degree Polarization	60
Appendix E. POE Code, excerpts for CP and V/H Polarization	86

Antenna Design, Modeling, and Testing on the WindSat Satellite Wind Direction Measurement System

I. INTRODUCTION

WindSat (Figs. 1-3) is a satellite-based multi-frequency polarimetric microwave radiometer being developed by the Naval Research Laboratory for the U.S. Navy and the National Polar-orbiting Operational Environmental Satellite System (NPOESS) Integrated Program Office (IPO). The primary objective of the WindSat mission is to demonstrate the capability of polarimetric microwave radiometry to measure the ocean surface wind vector from space.

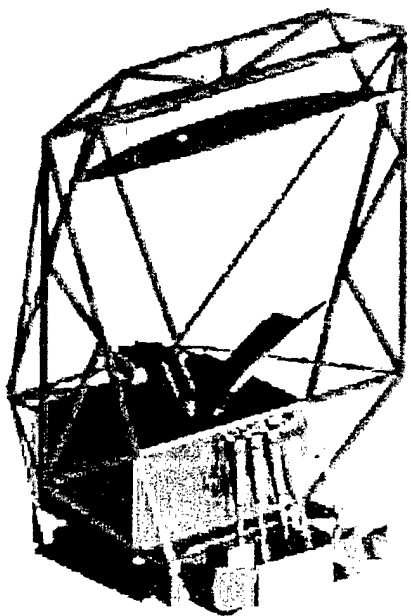


Fig. 1 Windsat system antenna structure

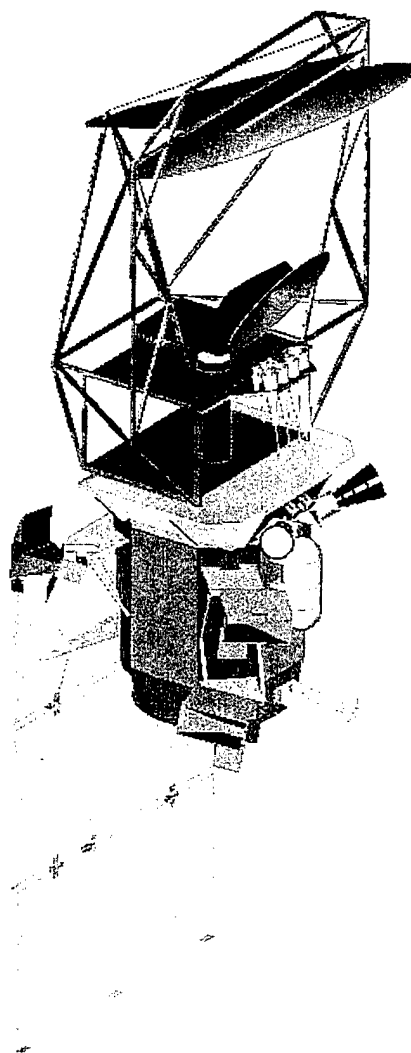


Fig. 2 Windsat antenna structure on spacecraft

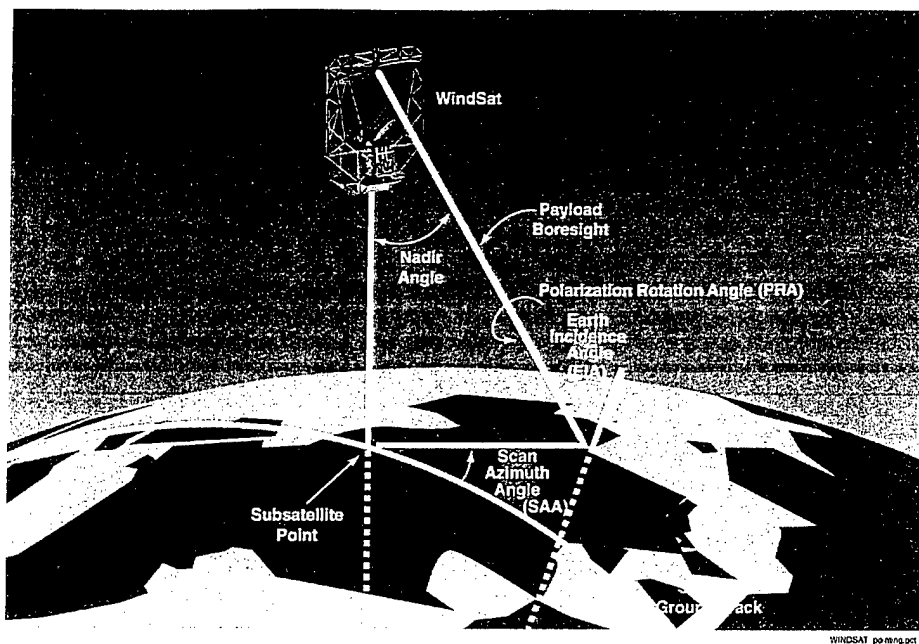


Fig. 3 Windsat system on orbit

The global ocean surface wind vector is essential information for short-term weather forecasts and warnings, nowcasting, and climatology and oceanography studies in both the civilian and military communities. The sensor also provides risk reduction for the IPO development of the Conical Microwave Imager Sounder (CMIS), which will be the next generation operational microwave imager. This risk reduction takes the form of technical lessons learned during the development of WindSat, algorithms and modeling and actual radiometer data. WindSat is the primary payload on the Air Force Coriolis satellite, which is scheduled to launch in early FY03.

WindSat will indirectly measure the ocean surface wind speed and direction using polarimetric microwave radiometry. This new technique measures the radiometric Stokes vector, which is expressed as

$$I_s = \begin{bmatrix} I \\ Q \\ U \\ V \end{bmatrix} = \begin{bmatrix} T_v \\ T_h \\ T_{45} - T_{-45} \\ T_{lc} - T_{rc} \end{bmatrix} = \begin{bmatrix} \langle E_v E_v^* \rangle \\ \langle E_h E_h^* \rangle \\ 2 \operatorname{Re} \langle E_v E_h^* \rangle \\ 2 \operatorname{Im} \langle E_v E_h^* \rangle \end{bmatrix} \quad (1)^*$$

*symbols explained in Appendix A

The microwave emission and scattering of the ocean surface vary with the wind speed and direction. By measuring these variations, WindSat data will be used to derive the ocean surface wind vector. Aircraft-based systems have successfully demonstrated this capability [1]. Currently, the Special Sensor Microwave/Imager (SSM/I) spaceborne radiometers operationally determine the wind speed only with accuracy better than ± 2 m/s [1]. An overview of the characteristics of radiometric performance is given in Appendix A as well as in referenced articles [2-10].

The performance requirements of the WindSat system are to measure the ocean surface wind speed and direction with an accuracy of ± 2 m/s and $\pm 20^\circ$, respectively. The system is designed to have a horizontal spatial resolution of ~ 25 km. This report will describe the design, modeling and calibration of the antenna system used to support these requirements.

II. Windsat Reflector Design

IIa. Windsat Reflector Design

The Windsat reflector design consists of a set of 11 clustered horns feeding a 72 in. diameter offset reflector. The offset configuration was used to minimize crosspolarization due to the struts and feeds blocking the main beam. The reflector size was limited by what would fit in the launch cannister. The focal length of 61.6 in., offset of 14 in., and spin axis offset of 15 in. were limited by the cannister dimensions and by preventing blockage of the main beam by the feed bench. A long focal length was preferred to minimize the reflector crosspol. Fig. 4 shows the design. The extremely low weight 16 lb. reflector was built by Composite Optics, Inc. (COI) out of composite material with a honeycomb rib backing, Figs. 5 and 6. The composite material was necessary to reduce thermal-induced roughness on-orbit as well as to maintain a low weight. The reflector spins at a rate of 30 rpm (revolutions/minute), so the low weight was needed to reduce the moment of inertia. The truss (manufactured at NRL) was also built from low coefficient of thermal expansion (CTE) composite material, to reduce on-orbit thermal-induced movement of the feeds relative to the reflector. Warm and cold loads scan over the feeds once per rotation to calibrate the receiver.

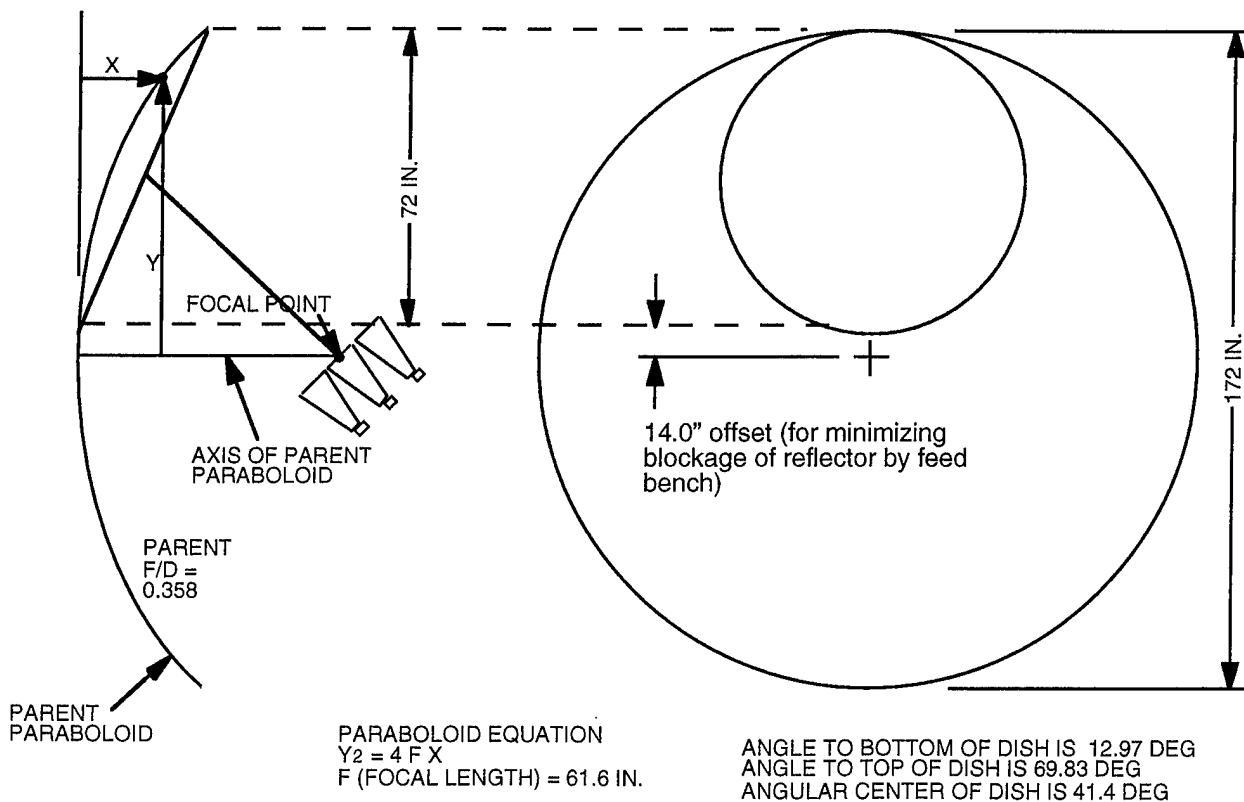


Fig. 4 WINDSAT reflector design. Spin axis offset is 15 inches

The reflector boresight is pointed down from the spacecraft at an angle of 45 degrees, resulting in an earth incidence angle (EIA) of 53.04652 deg. for the 37 GHz circular polarization (CP) focus feed.

Several feed configurations were initially considered. The optimum configuration would be to use one broadband multi-polarized feed at the focal point. This technique was discarded due to the impracticality of developing a feed with low enough crosspol necessary to achieve a good Stokes coupling matrix. Also, a broadband feed would be noisier with out-of-band signals interfering with the desired signals. It was necessary to keep the three polarizations of each frequency at the same EIA, but the separate frequencies could be at separate EIA values. A stacked set of feeds was decided on to optimize the feed performance for each separate polarization and frequency. The 37 GHz set of feeds were placed with the center CP feed at the focal point. It was decided to place the CP feed at the focal point (the optimum position) rather than the 37 GHz 45 feed purely on mechanical packing considerations. The 10.7 GHz feeds were placed above the 37 GHz feeds so that their wider beams would fall completely on the earth. If they were placed below the 37 GHz feeds, their beams would be at a greater EIA and fall partially off the earth. A range test experiment done to measure the Stokes matrix with a feed alone and with feeds close by demonstrated that mutual coupling between the feeds was minimal.

The Ohio State Reflector Code (OSUREF) and GRASP8w reflector modeling software (commercial-off-the-shelf (COTS) from Ticsra, Inc.) were used for all the reflector modeling. Appendix B gives information regarding these code. The codes gave almost identical results for the modeling (see Appendix C), however, each code had strengths and weaknesses, making it preferable to use one or other for each of the modeling tasks. The GRASP8w software was, in general, more flexible for modeling the multiple feed offset design, and also was able to match the range geometry exactly, so was the primary code used for the work.

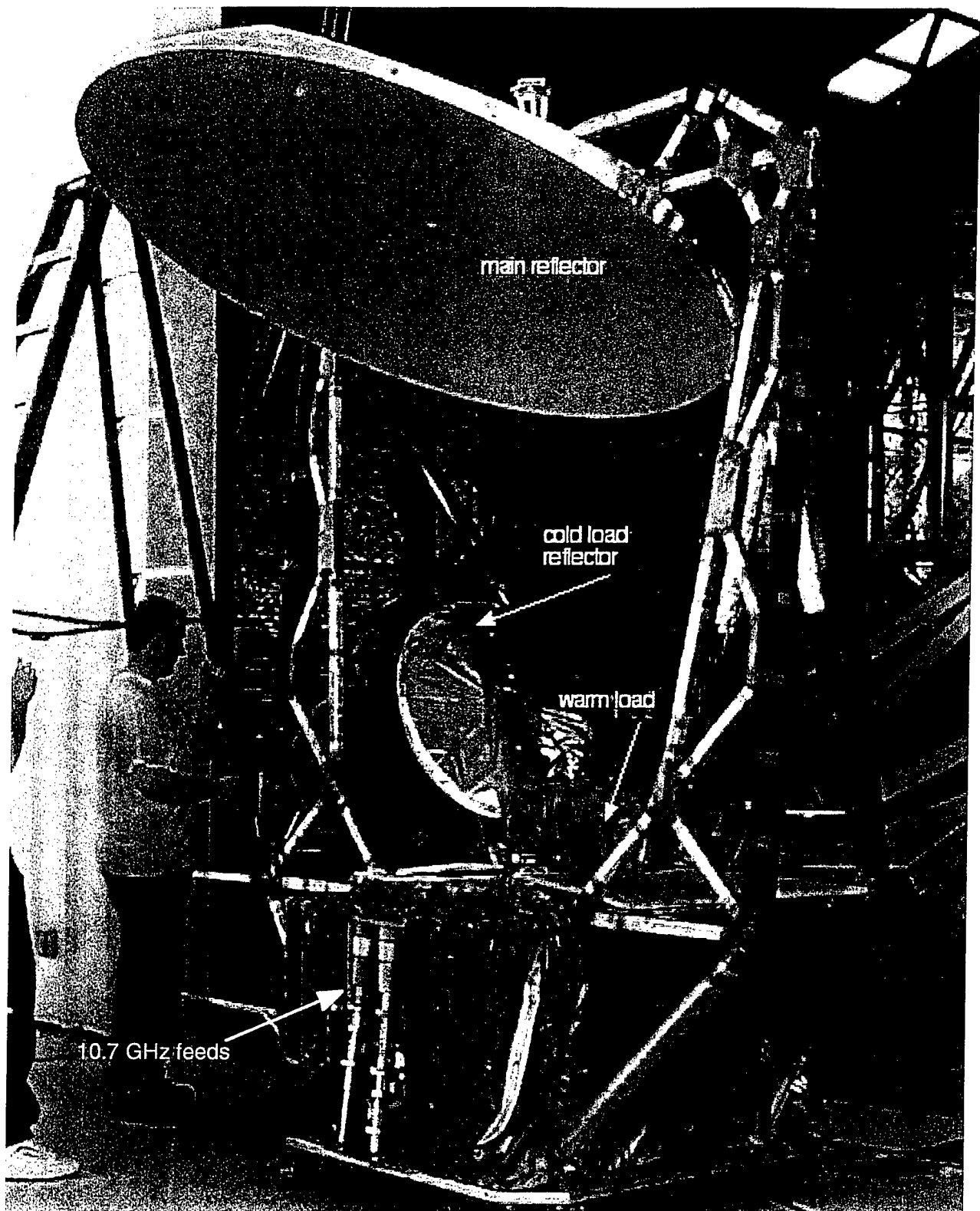


Fig. 5 WINDSAT flight antenna, with truss, reflector, and feed bench (getting ready to be tested on range). Warm and cold load appear below the reflector.



Fig. 6 WINDSAT prototype reflector built by Composite Optics, Inc., showing backside rib structure

IIb. Windsat Reflector Surface Distortion

The reflector had a surface roughness of 2 mil rms (built to a 3 mil spec). Fig. 7 shows the surface roughness of the reflector as mapped with precision theodolites. This data was input into both the OSUREF or GRASP8w software to predict the effects on the gain, pointing, and the Stokes coupling matrices. These predictions were also compared to values computed by the Ruze equation. The Ruze [11] equation predicts the decrease in beam efficiency of a reflector with rms surface errors. The Ruze equation is:

$$G = G_0 [e^{-(4\pi\sigma/\lambda)^2}]$$

where σ = rms surface tolerance in meters
 λ = wavelength in meters.

The OSUREF code was used to verify this equation by using photogrammetry data artificially 'roughened'. As shown in Table I, the OSUREF code predicts values just slightly higher than the Ruze equation predictions.

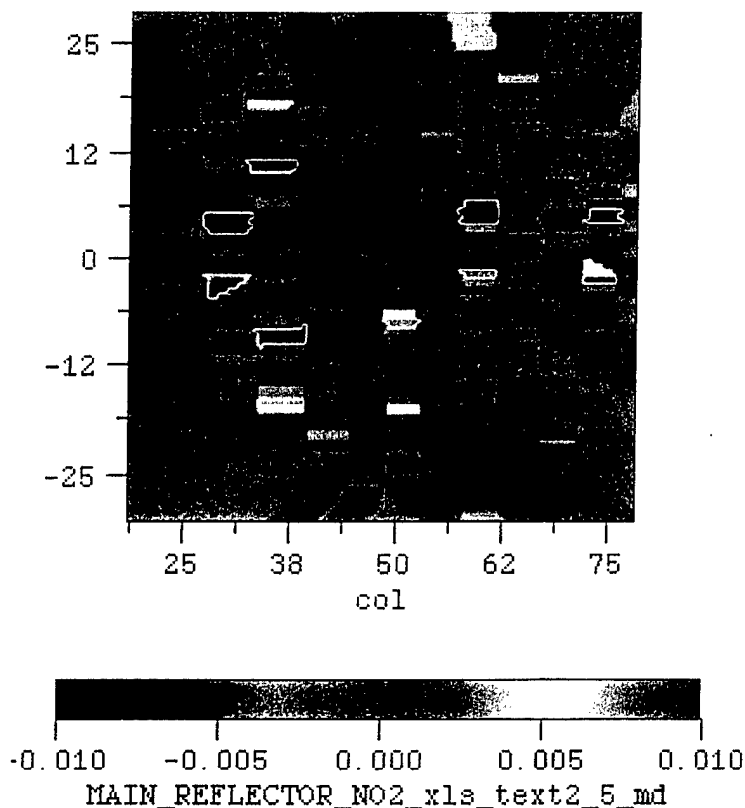


Fig. 7 Surface roughness of WINDSAT reflector (breadboard unit) in inches taken by photogrammetry data

Table I RMS Surface Distortion vs. Beam Efficiency as Predicted by the OSUREF code and the Ruze Equation for Reflector at 37 GHz

RMS Surface Tolerance (λ)	RMS Surface Tolerance (mil)	Peak Gain (dB)	OSUREF Predicted Efficiency (%)	Ruze Equation Predicted Efficiency (%)
0.0	0.0	54.81	100.0	100.0
0.0094	3.0	54.75	98.63	98.62
0.0125	4.0	54.70	97.60	97.55
0.0157	5.0	54.65	96.38	96.18
0.022	7.0	54.49	92.90	92.64

Thermal and gravity gradient distortion was also examined. Using a finite element model with 4500 points representing the composite WINDSAT reflector surface (Fig. 8), seven thermal and gravity distortion cases were analyzed representing worst case on-orbit conditions. The resulting surface distortions were then input into the OSUREF code and the impact on the reflector

performance analyzed. All cases had mean and rms surface tolerances of less than 1.5 mil, which had minimal impact on the reflector efficiency and angular pointing offsets of less than 0.005 deg.

It was noticed that the surface distortions due to the thermal and gravity gradients had less effect on efficiency than would be predicted from a straight rms distortion prediction. This is because the distortions remain constant over large portions of the dish (Fig. 9) and result in beam movement rather than beam spreading and resulting loss of efficiency. To exemplify this, the distortions due to gravity gradients were multiplied to provide an overall rms surface accuracy of 6 mil. The efficiency for this case was 98.0% compared to 94.5% for a random rms surface accuracy of the same 6 mil.

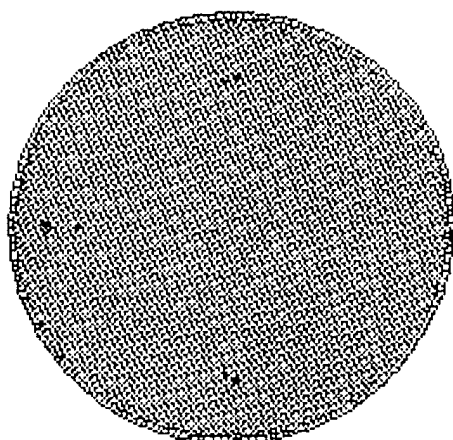


Fig. 8 Finite Element Grid of Windsat Reflector



Fig. 9 Symbolic diagram of (a) magnified rms distortions and (b) thermal or gravity gradient reflector distortion

IIc. Beam Efficiency

The overall beam efficiency goal was 95%. Factors other than surface roughness contributing to beam efficiency were cross-polarization, beamwidth, and struts. Table II lists the predicted beam efficiencies for the worst-case 37 GHz frequency band, which meets the system requirement. Section IV on range data shows beam efficiencies derived from the range data, which correlate well with the predictions.

Table II Beam Efficiency
(found by multiplying individual efficiencies together)

	Efficiency %
Cross-polarization	99.1
Struts	99.6
2.5 Beam-widths	99.2
Surface Roughness (4.2 mil)*	97.3
Total:	95.3

*The surface roughness is the rms roughness for 3 mil manufacturing and 3 mil for on-orbit thermal roughness.

IId. Struts

The contributions from struts on Windsat reflector performance was modeled by the OSUREF. The Windsat truss was designed to keep all struts out of the main beam with a 6 inch clearance. The focused feed case was used. The OSUREF code models reflector-strut and feed-strut scattering using an aperture integration technique (as opposed to a physical optics technique, such as used in the GRASP8w code, which is more computationally intensive). To reduce the model complexity, the struts were modeled with circular cross-section rather than their actual square cross-section. Fig. 10 shows two views of the model (using flat plate for reflector for visualization purposes only) using 24 struts. The struts are shown in these visualizations as lines without thickness. [12]

The modeling was done to predict the strut effects on beam and crosspol efficiency, as well as the impact on the Stokes parameters. The efficiency modeling was done at 37 GHz, which has the tightest efficiency margin. The Stokes impact modeling was done at 10.7 GHz. The 10.7 GHz beam is the broadest, so should be affected the most with any strut impact on the Stokes parameters.

The cross-pol beam efficiency was reduced by the effect of the struts as well as the overall efficiency. The results are listed in the previous section, Table II.

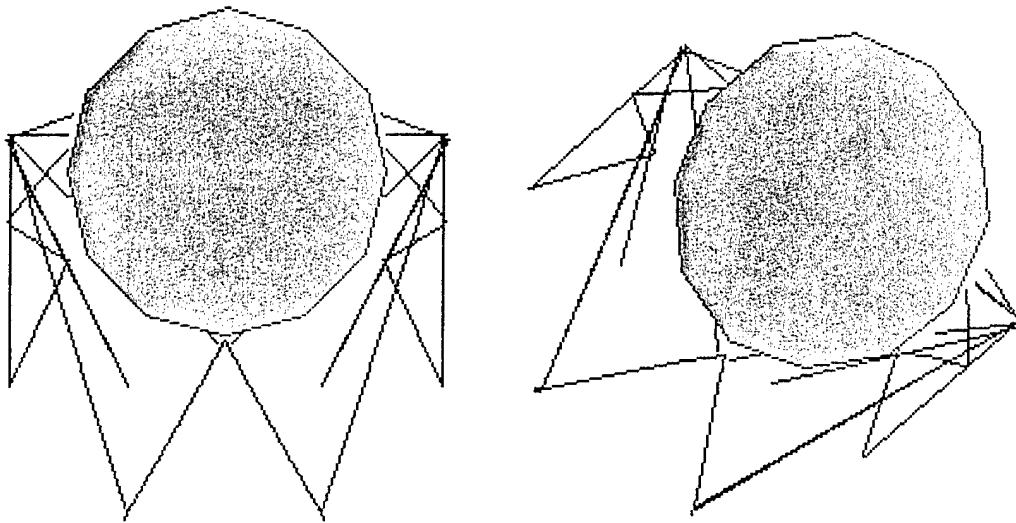


Fig. 10 Visualization of Windsat Reflector with 24 struts

The OSUREF code was modified to allow it to print out data in a format suitable for Stokes computation with a modified Poe code (see Appendices A, D, and E). A matrix of 71 x 71 points was used (similar to range data). The OSUREF code outputs polarizations either in co/cross or theta/phi polarizations, so the polarizations needed to be transformed into the vertical/horizontal polarizations needed by the Poe code.

When a symmetric system of 24 struts was used, the Stokes terms remained very low (as if no struts were present), presumably because the effect was symmetric and canceled out. The runs were repeated using only one half of the struts (on one side of the reflector). This is a worse case scenario with no canceling effects. For this case, with 1.5 inch struts, the 3rd and 4th Stokes terms increased from -45 dB to -38.5 dB (peak). When the struts were increased in size to 2.5 inch, the terms increased further to -32.6 dB (peak, for 3rd Stokes). The final size of the struts is 1.2 inch. These results indicate that the struts will have minimal impact on the system performance. Near-field effects of the horn cluster and ground plane will also impact the Stokes parameters, but cannot be modeled by the OSUREF or the Ticra code. Field testing of a horn in the Windsat configuration with and without a set of horns nearby showed that coupling effect also to be minimal on the Stokes' matrix.

The strut effects on the crosspol patterns appeared fairly minimal in the center of the pattern, so that it is not expected that the struts would have a

large effect on degrading the Stokes terms. Figs. 11 & 12 show a comparison of the co- and cross-pol patterns with and without 24 struts for the focused case (10.7 GHz).

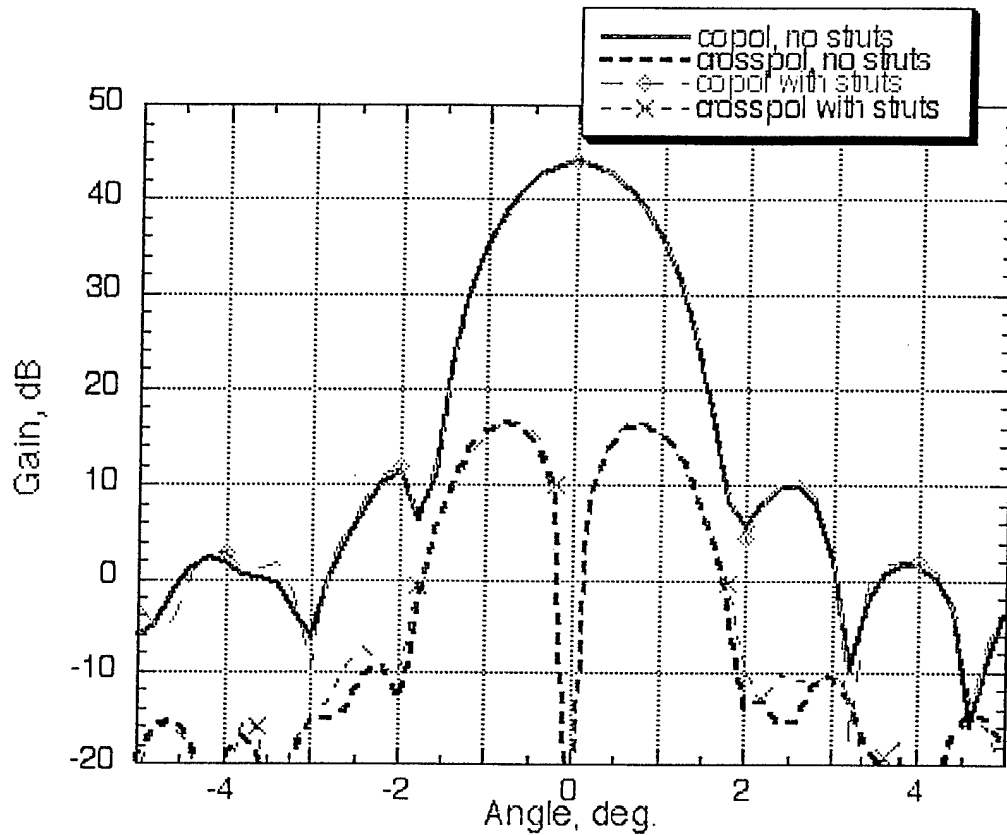


Fig. 11 Copol and crosspol patterns of focused case (10.7 GHz) 0 deg. cut, with and without struts (using 24 symmetric struts)

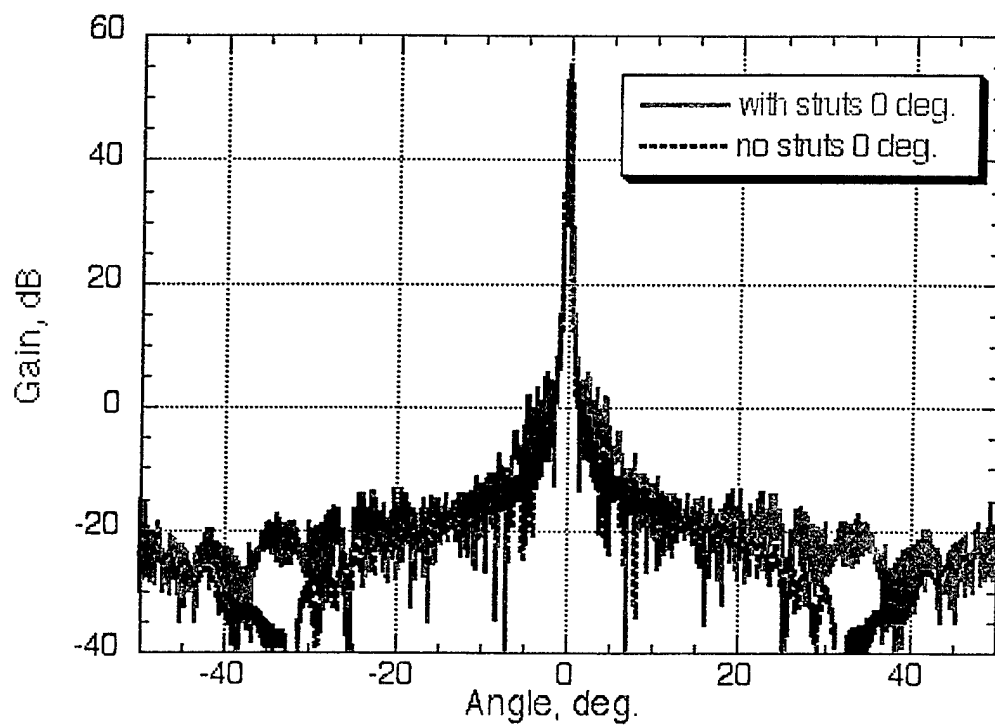


Fig. 12 Copol patterns of focused case (10.7 GHz) 0 deg. cut, with and without struts (24 symmetric struts)

III. Horn Design & Measurement

Circularly symmetric corrugated horns were used to feed the reflector. These horns provide good symmetry between the e-plane and h-plane cuts, low cross-polarization, low sidelobes, and wide bandwidth. Table III lists the feeds with their positions relative to the feed bench as well as beam positions. Table IV lists the horn frequencies and their bandwidths.

TABLE III Horn positions and respective beam positions and toe-in angles for 14" offset reflector, 72 inch diameter, with 61.6 inch focal length

Freq., GHz	Polarization	Position x, inches	Position y, inches	Position z, inches	Elevation angle (x) from boresight deg.	Azimuth angle (y) from boresight deg.	Toe-In Angle* deg.
6.8	VH	.2062	9.9630	.2603	-0.1678	8.0555	-7.892
10.7	VH	3.1369	-4.4921	-.0330	-2.5514	-3.65122	+3.713
10.7	CP	2.9947	0.0	-0.1823	-2.4359	-0.0	0
10.7	45	3.1369	4.4921	-.0330	-2.5514	3.65122	-3.713
18.7	VH	-2.1571	-2.7115	-0.0127	1.75514	-2.2058	+2.083
18.7	CP	-2.2067	0.0	-0.2155	1.7955	0.0	0
18.7	45	-2.1571	2.7115	-0.0127	1.75514	2.2058	-2.083
23.8	VH	0.4475	-8.1021	0.4134	-3.6414	-6.5655	+6.444
37.0	VH	0.0344	-2.2488	0.0359	-0.028	-1.8297	+1.789
37.0	CP	0.0	0.0	0.0	0.0	0.0	0
37.0	45	0.0344	2.2488	0.0359	-0.028	1.8297	-1.789

* Positive angle means feeds rotated clockwise looking from behind feed towards reflector

Table IV Feed bandwidth, beamwidth, and desired residual Stokes Coupling

Freq. (GHz)	Bandwidth (MHz)	Secondary Pattern Beamwidth (deg.)	Residual Stokes Coupling % (dB)
6.8	125	1.9	0.5% (-23 dB)
10.7	300	1.2	0.1% (-30 dB)
18.7	750	0.68	0.1% (-30 dB)
23.8	500	0.53	0.5% (-23 dB)
37	2000	0.34	0.1% (-30 dB)

Figure 13 shows the actual WINDSAT flight feeds. Fig. 14 is a CAD diagram of the feeds in the flight bench.

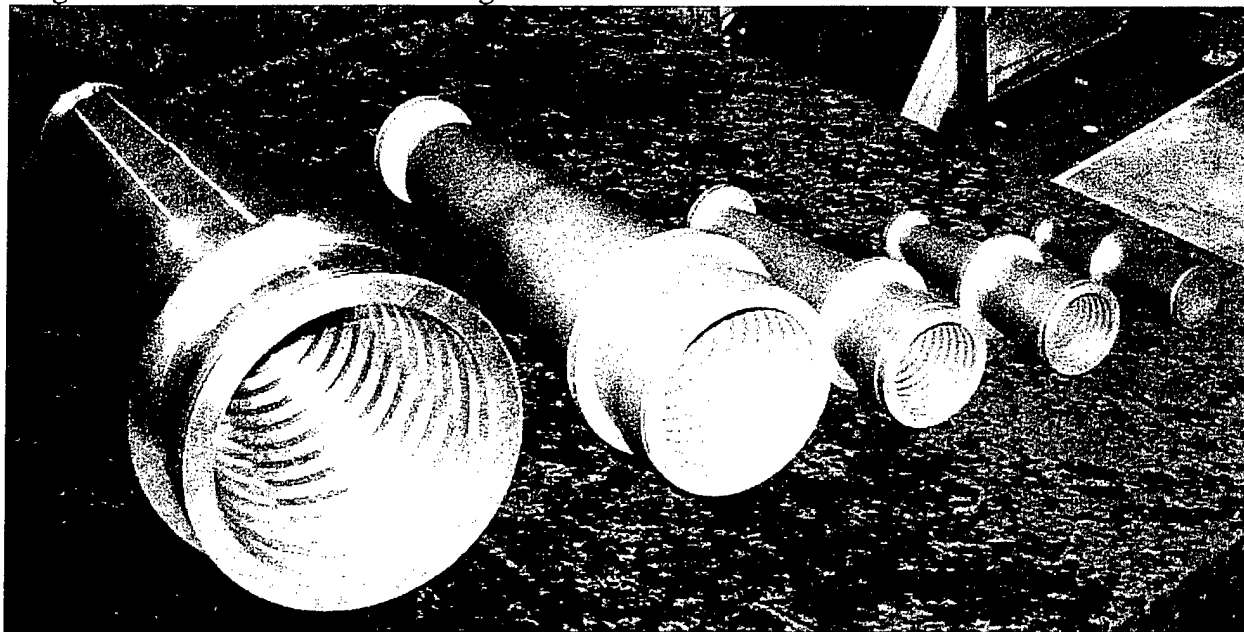


Fig. 13 Corrugated horns, left to right, 6.8, 10.7, 18.7, 23.8, and 37.0 GHz

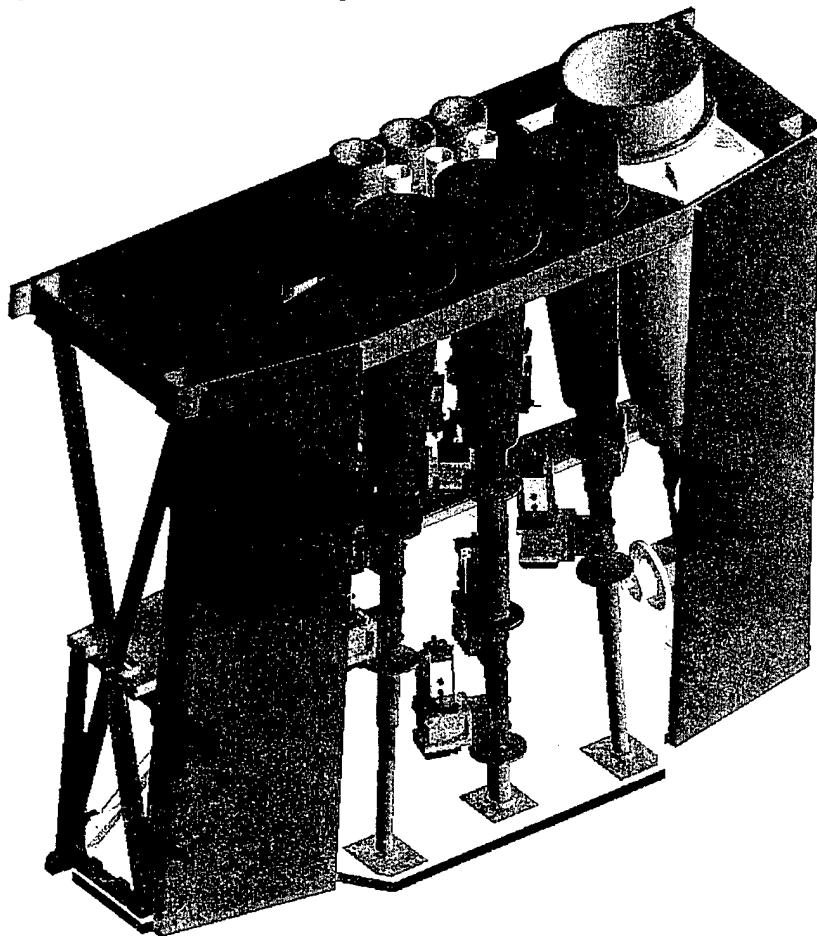


Fig. 14 Windsat feed bench

Figure 15 shows a diagram of the ortho-mode transducer (OMT) assembly. Fig. 16 is a diagram of the basic corrugated horn feed design. Fig. 17 shows a photo of the flight feeds installed in the feed bench. Fig. 18 shows the feed bench coordinate system. Fig. 19 shows the OMT Orientation looking from the reflector towards the feeds.

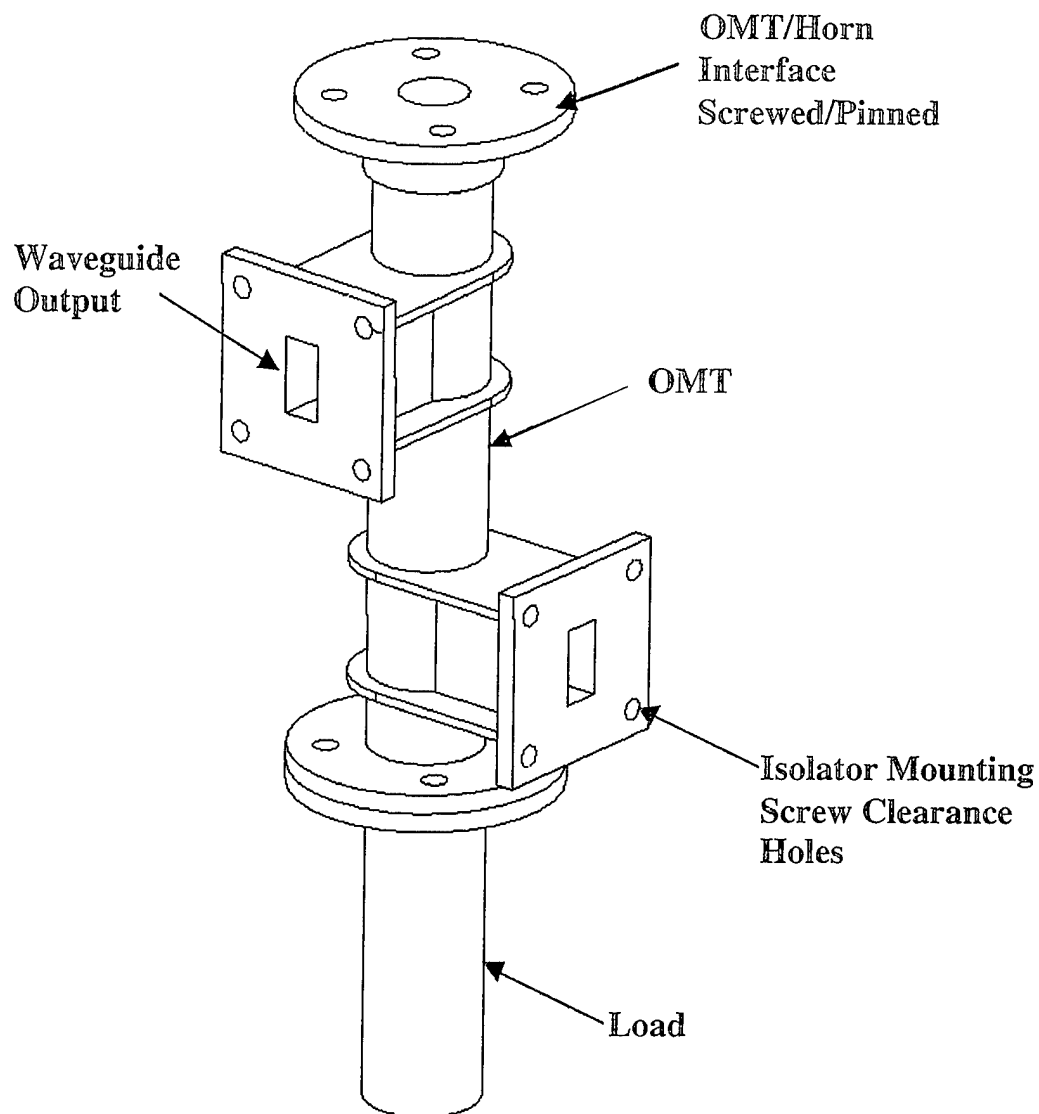


Fig. 15 OMT assembly

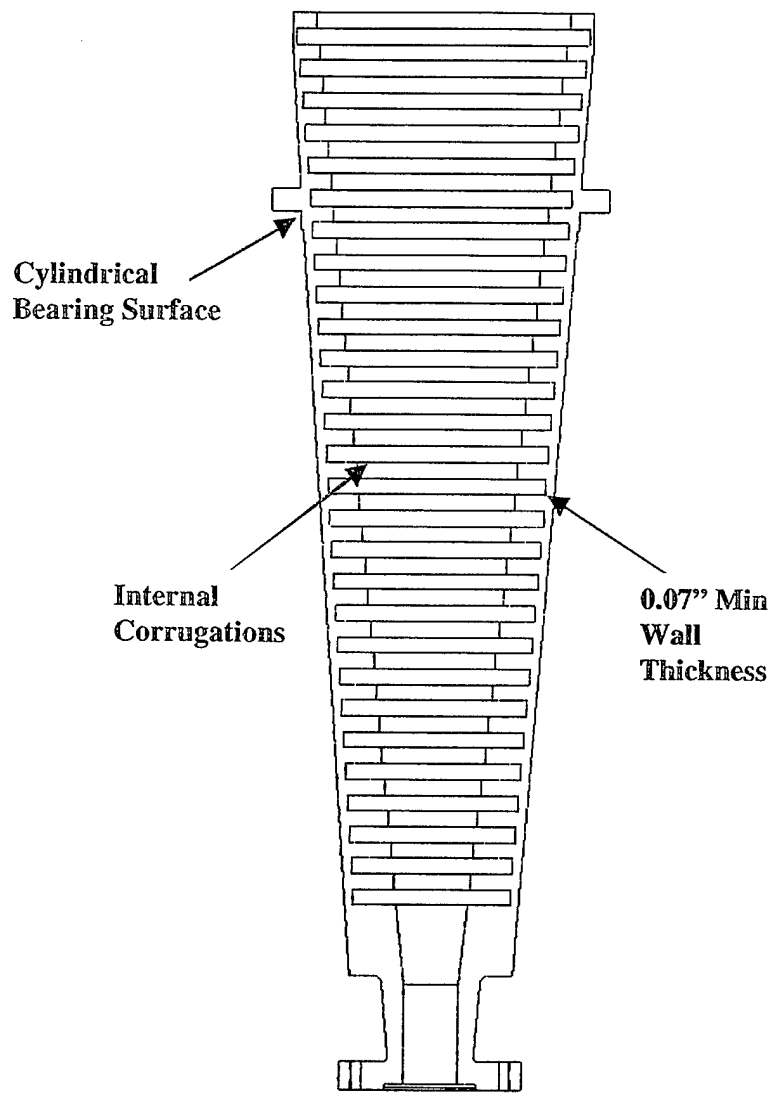


Fig. 16 Basic Corrugated horn feed design

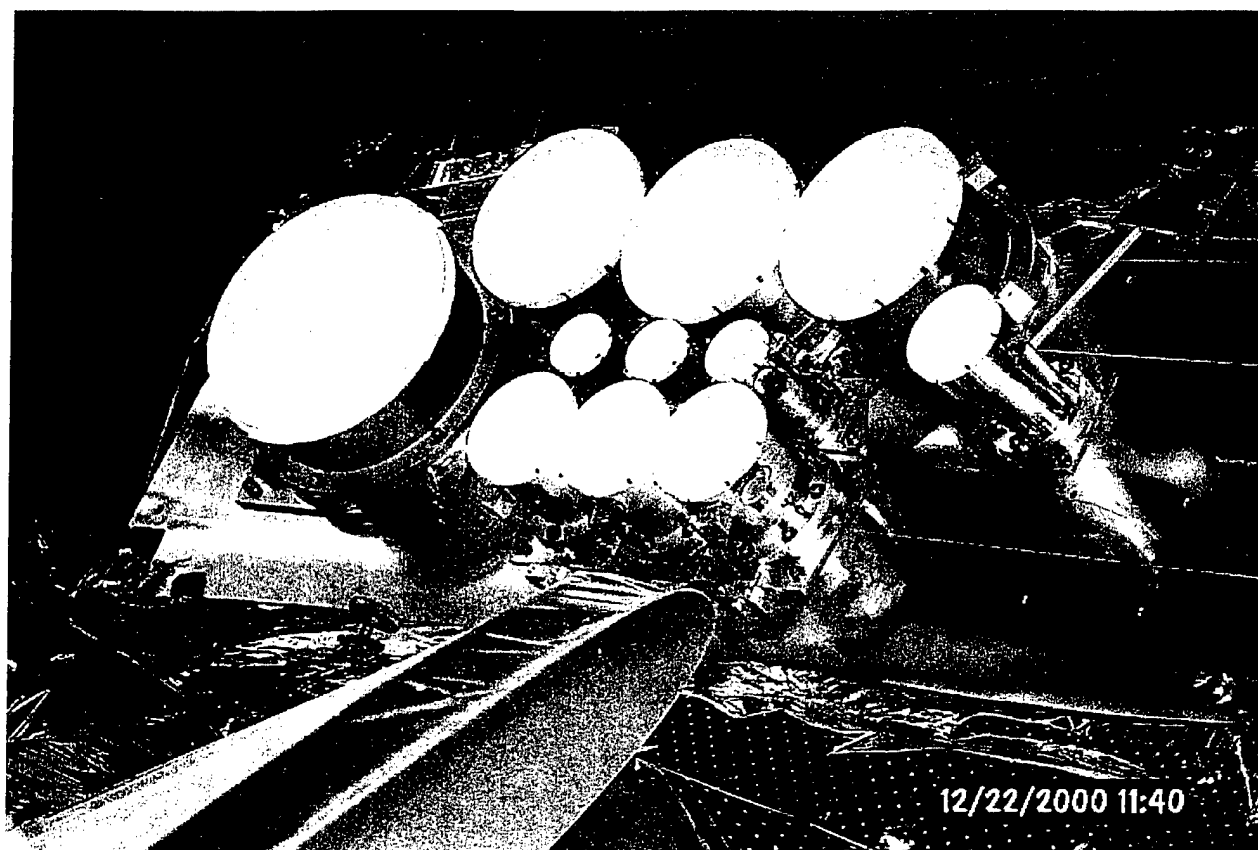


Fig. 17 Flight feeds installed in the feed bench

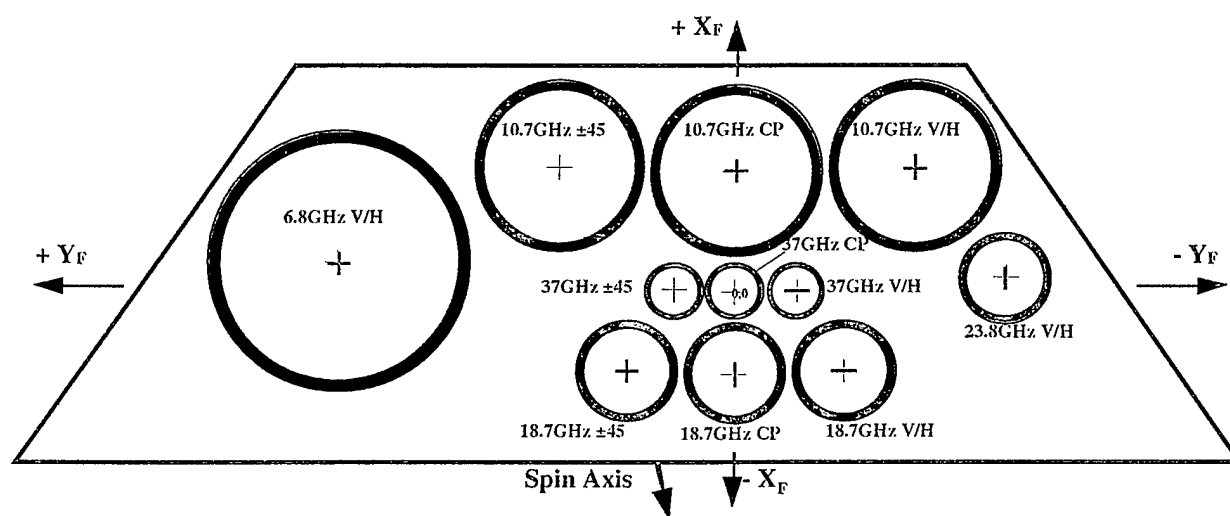
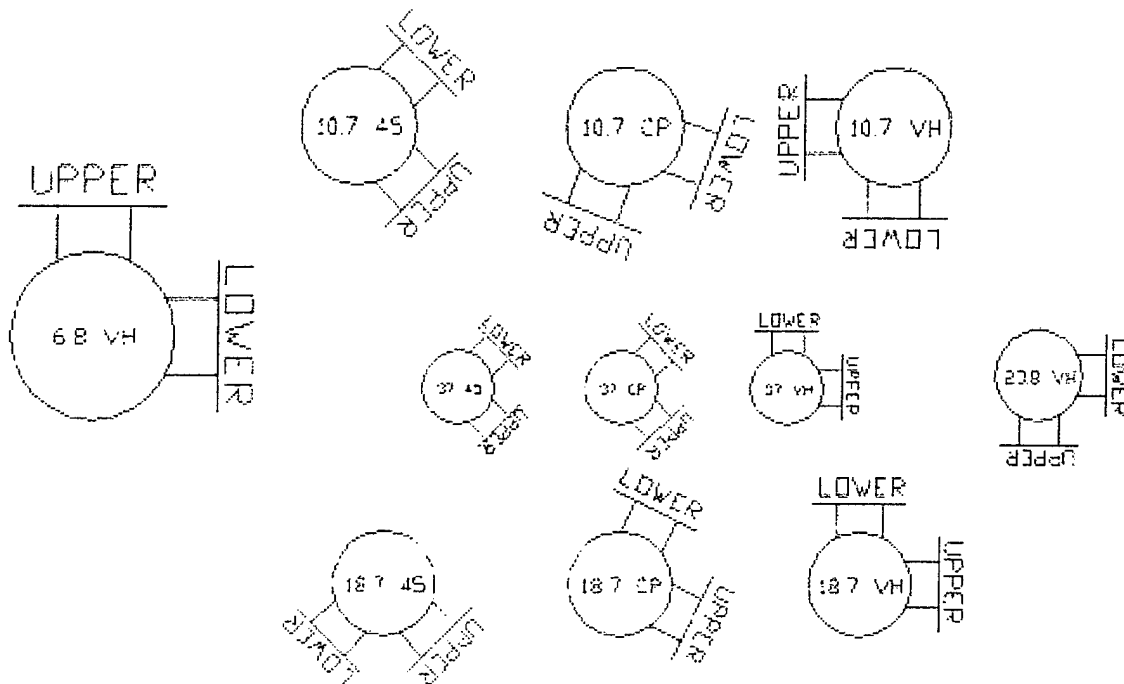


Fig. 18 Feed bench coordinate system



FEED BENCH VIEW FROM REFLECTOR

Fig. 19 OMT Orientation looking from the reflector towards the feeds

The CORHORN program from Antenna Software Limited was used during the WINDSAT feed horn design to predict the performance of various designs. The CORHORN program models conical corrugated horns exclusively. The program uses a modal matching technique to predict the radiation characteristics and input impedance.

For the feed design, high beam efficiency was obtained with a -26 dB feed illumination taper. Corrugations were designed to yield low feed cross-polarization (cross-pol), and a gradual feed taper over the length yielded low voltage standing wave ratio (VSWR). Since the requirement for beam width was the same for all the horns, the same design, scaled with frequency, was used for all the horns except for the 6.8 GHz horn, which was shortened to reduce size and weight. This was acceptable because it was not a polarimetric horn.

Fig. 20 shows an example of the two-dimensional (2D) corrugated design used as input to the CORHORN program. Fig. 21 shows the excellent correlation between CORHORN model predictions and anechoic chamber data for the co-polarization (co-pol). The chamber had a reflectivity of -45 dB, not low enough to measure the cross-pol performance of the feed horns, which had a predicted peak cross-pol of -44.2 dB in the 45 deg. plane. With a chamber reflectivity of

-45 dB, this would result in an inaccurately measured cross-pol of -38.6 dB for the Windsat feed horn (adding voltages), which is close to that observed.

The CORHORN program feedhorn predictions were used in all the Ticura modeling of the Windsat reflector. The CORHORN cross-pol peak was set higher to -39 dB by artificially adding 5.2 dB. This was done because it was assumed that the cross-pol of an actual feed would be slightly higher than predictions due to machining tolerances. Also, early Ticura simulations which correlated squint in the reflector crosspol to the feed cross-pol indicated a -39 dB feedhorn cross-pol. Also, some feedhorn measurements done on the Windsat range in Florida (-55 dB reflections) with a preliminary feed indicated a cross-pol of -39 dB.

The fact that the Grasp8w simulations were not done with actual feed data can account for some of the small differences in the modeling and predictions. Other factors include near-field interactions between the feed and dish and interactions between the feed and the other 10 feeds in the feed bench.

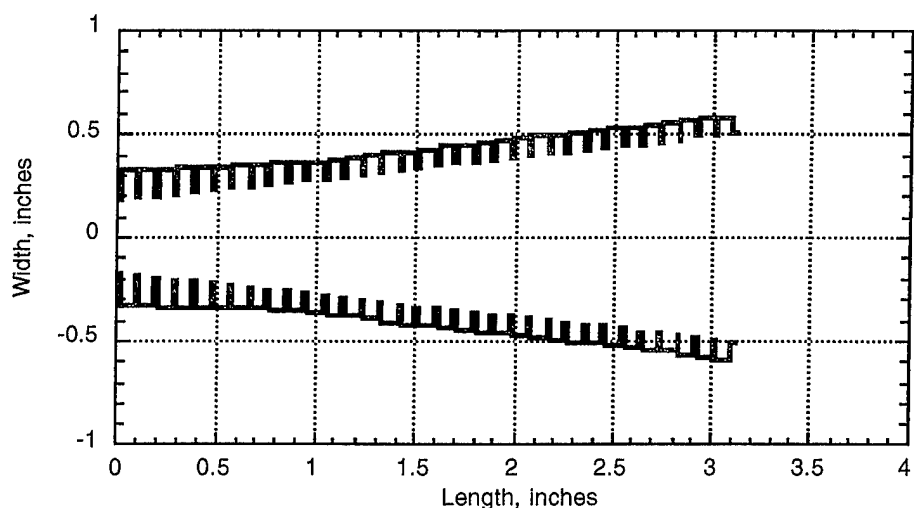


Fig. 20 Example of 2D CORHORN input

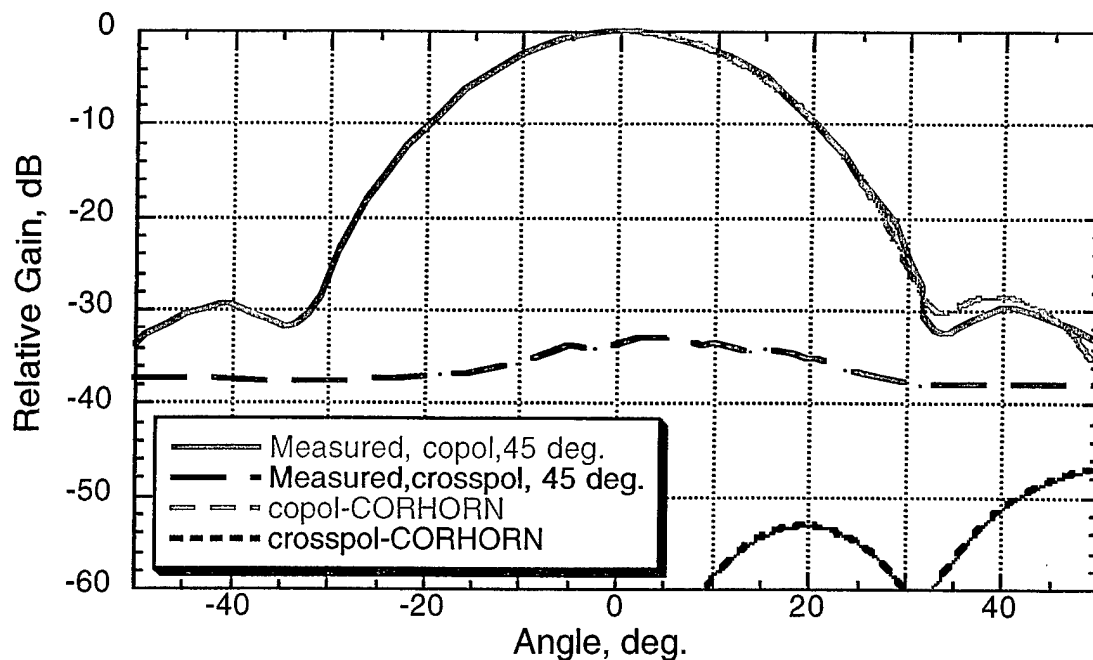


Fig. 21 Comparison of measured and CORHORN modeled feed data, showing excellent correlation of co-pol pattern, 37 GHz feed with 6° flare angle

The feed insertion loss requirement was 0.40 dB for the 37 GHz feed and 0.30 dB for the other frequencies. Table V lists the feed/OMT losses per band, which meet the requirement. The 37 GHz losses are higher due to its small size and the greater impact of the machining tolerances on its performance.

Table V Average Feed/OMT Loss Per Band, dB

Antenna feed/OMT Loss (dB)	6.8	10.7	18.7	23.8	37
Front Port Dual Linear Feed	0.03	0.04	0.06	0.08	0.16
Rear Port Dual Linear Feed	0.04	0.05	0.08	0.11	0.23
Front Port Dual Circular Feed		0.07	0.11		0.33
Rear Port Dual Circular Feed		0.08	0.13		0.39

The horns needed covers to avoid collection of debris during launch and to keep the sun from heating the inside cavity. Beta cloth (Teflon impregnated fiberglass) was initially used as a cover, but was found to adversely affect the feed isolation for the upper frequencies, resulting in large swings in the Stokes vector over frequency. Though this could be calibrated out with difficulty, it was decided to change the feed cover material to $1/2 \lambda$ teflon for the 10.7, 18.7, 23.8 and 37 GHz horns. Fig. 22 shows the feed isolation for the 37 GHz horn with and without the beta cloth compared to the requirement of -57 dB. Fig. 23 shows the feed isolation with the teflon material.

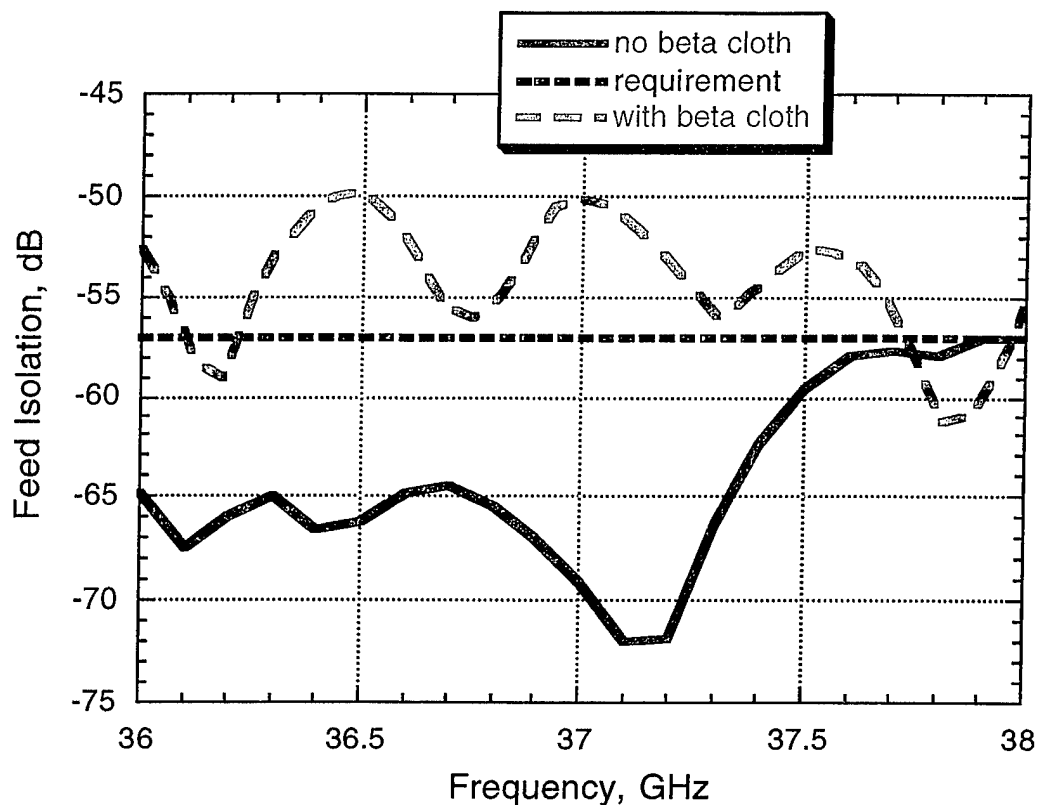


Fig. 22 Feed isolation with and without beta cloth for 37 GHz horn showing adverse affect of beta cloth on isolation (due to high reflections)

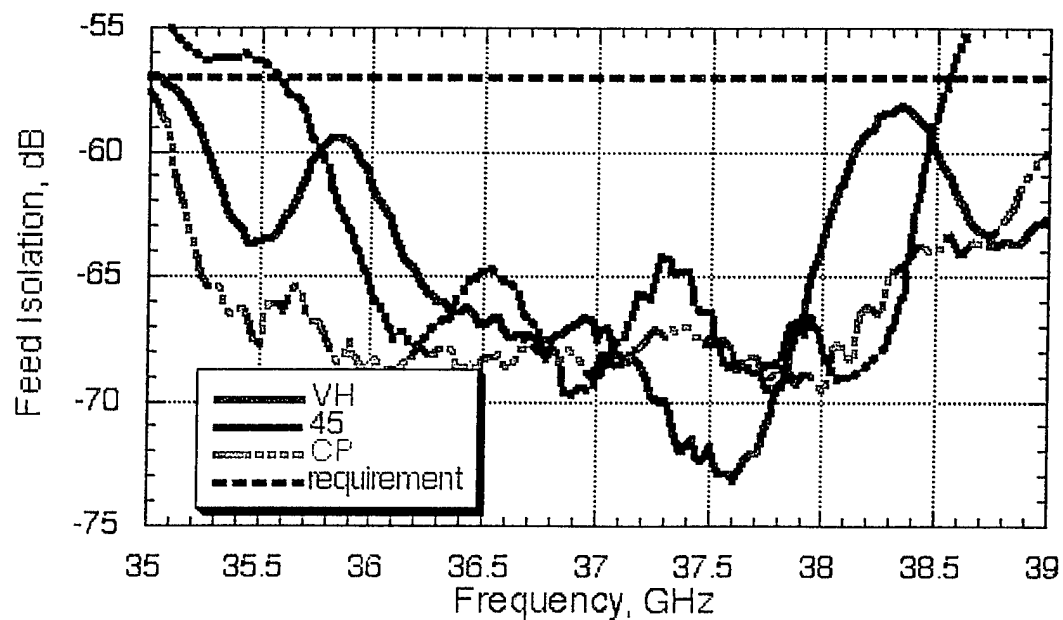


Fig. 23 Feed isolation with teflon covers, 37 GHz Flight Feeds (Includes Horn And OMT's, No Polarizer), Requirement is over 36-38 GHz

IV. Range Measurement

IVa Windsat Test Range:

Testing for the Windsat reflector was conducted at the NRL Antenna Test Range located in Melbourne, Fl. Operations were conducted by Microstar, Inc. The range consists of a building housing the test and reference antennas along with test equipment and computers (Fig. 24) and a set of source antennas mounted on a 500 ft tower (Fig. 25).

The source transmit antennas (3 bands) were mounted on a tower located 440 ft above the test antenna over a slant range of 1317 ft as determined by theodolite laser ranging to a prism mounted on the antennas. The source antenna designs were symmetrical parabolic reflectors with wideband feeds. A polarization grid (waveguide below cutoff orthogonal to the feed polarization) is located in the aperture plane and provides purity of the each antenna to -80 dB. Three parabolic antennas of similar design, but different size (0.5, 1.0, and 2 feet) cover the bands from 6.8 to 37 GHz. Each antenna mechanically rotates 90 degrees \pm .003 degrees. The 90 degree rotation accuracy is critical to obtaining Windsat polarization purity calibration data.

Figure 26 shows the critical range geometry axis and distances. The elevation axis is approximately 5 ft below the vertex of the reflector. When raster cuts are obtained, there is approximately 377 degree of phase per degree of elevation change due to this offset at 37 GHz. A nodding toward or away from the tower source relative to the stationary reference antenna induces a phase ramp which needs to be removed to evaluate the elevation phase of the antenna. However, the elevation phase changes are not critical to the Stokes matrix calculations, because the Stokes power calculations are performed at each individual point in the pattern and then summed.

A detailed summary of the range testing can be found in a Microstar report [13].

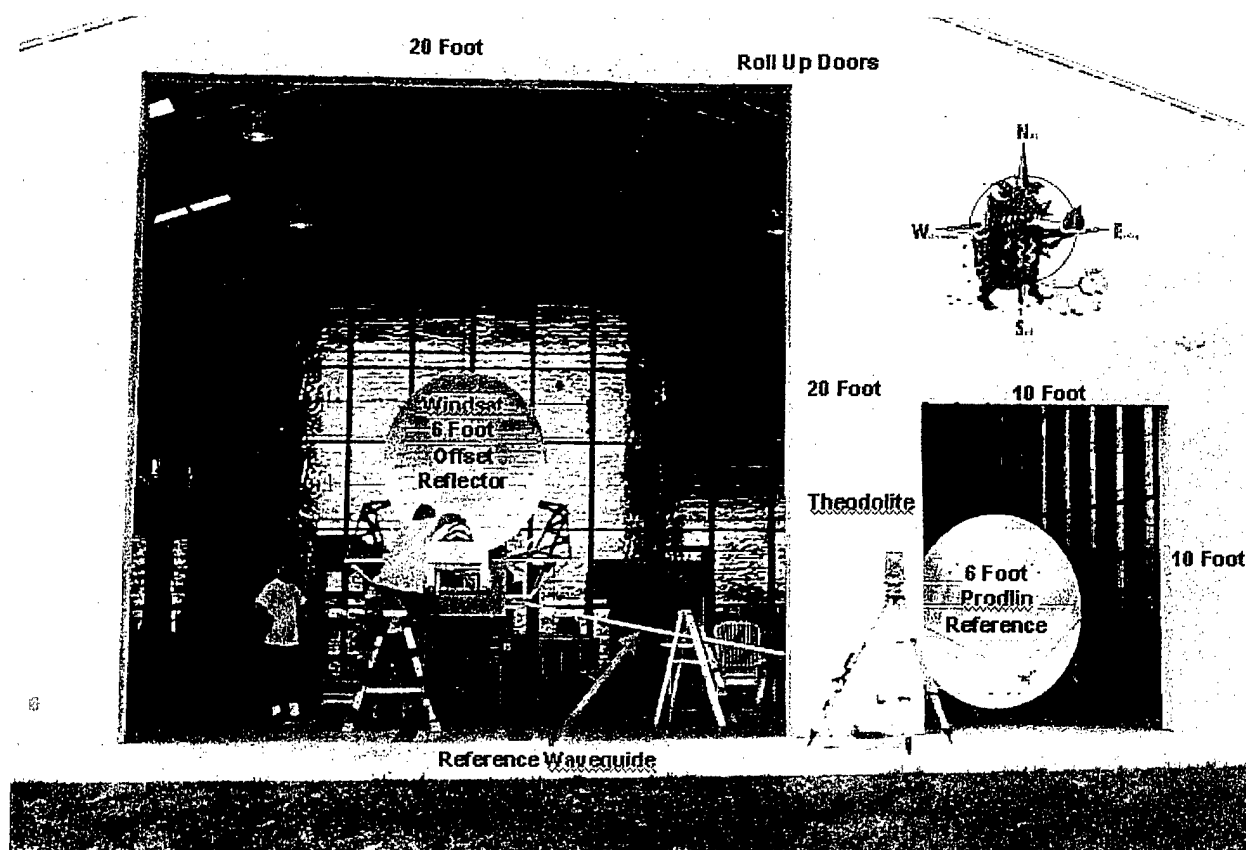


Fig. 24 Range setup showing Windsat reflector and reference reflector



Fig. 25 500 foot tower

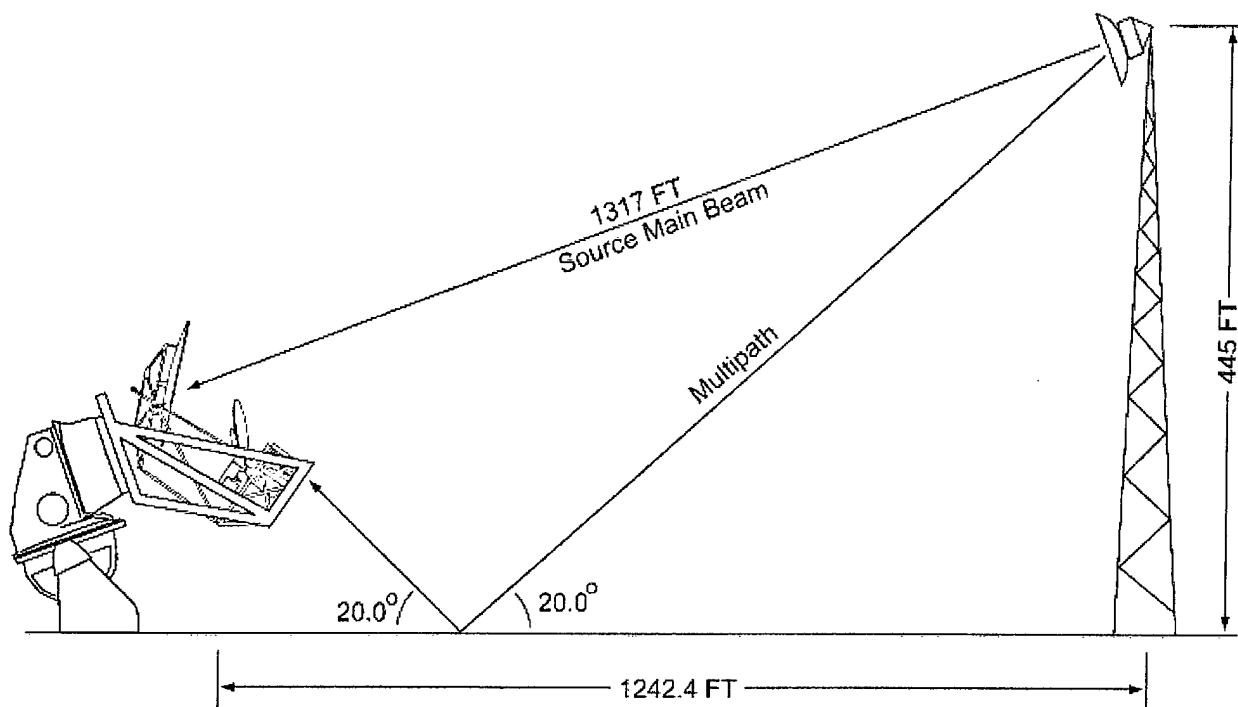


Fig. 26 Range Configuration

The vertical and horizontal polarized sources were aligned to an orthogonality of better than 0.010 degree. Rolling the Windsat feed located in the 72 inch offset reflector 90 degrees \pm .01 degree between the polarizations validated this alignment. Null depths less than -65 dB were obtained as the reference for polarization alignment. An 8510 Network analyzer with a dynamic range greater than 90 dB based on link budget and a measured approximate 10 dB SNR served as the measurement equipment for the alignment.

The Stokes processing code was modified to do an error analysis on the vertical and horizontal powers. The modeling (Fig. 27) showed that dynamic ranges on the order of -55dB were adequate for the Stokes evaluations, which required Stokes errors of less than -30 dB for the range measurements. The range exceeded this requirement.

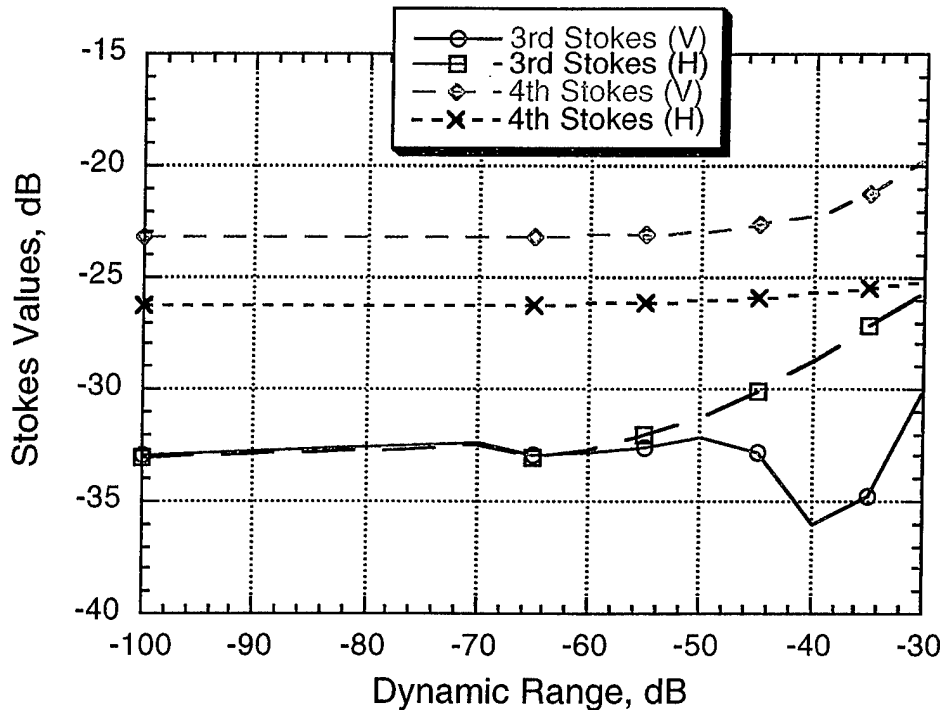


Fig. 27 Stokes values versus dynamic range for the 10.7 GHz focus case

IVb Tower Twist:

The transmit tower had three guy wires to minimize sway, but for the tight polarization rotation requirements on the Windsat systems, this was not sufficiently stable in twist. A system was developed to measure the twist and account for it in the measurements in the form of a tower twist file which took out the twist angle as a function of time during the measurements. This system involved recording movement of the tower via a remote camera located on the tower and viewing the Windsat reflector. Table VI shows a typical twist file. A new value for the twist is used for each of the 43 elevation cuts, which are taken sequentially during the testing. As can be seen, the twist varied by a few hundredths of a deg. during a typical measurement. The inclinometer alignment value (~ 0.3 deg.) for the tower polarizer is added to the twist for input to the main processing code.

Table VI Tower twist data for 37 GHz, full run, VH feed, 2/7/01

Elev. Cut #	Tower Twist, deg.	Elev. Cut #	Tower Twist, deg.	Elev. Cut #	Tower Twist, deg.
1	.3116	16	.3072	31	.3015
2	.3101	17	.3078	32	.3037
3	.3091	18	.3068	33	.3025
4	.3043	19	.3001	34	.3019
5	.3027	20	.3082	35	.3048
6	.3027	21	.3040	36	.3029
7	.3027	22	.3089	37	.3030
8	.3058	23	.3035	38	.3067
9	.3053	24	.3059	39	.3048
10	.3068	25	.3032	40	.3062
11	.3037	26	.3033	41	.3050
12	.3068	27	.3040	42	.3055
13	.3067	28	.2999	43	.2991
14	.3093	29	.3058		
15	.3062	30	.3037		

IVc Predicted versus Measured Gain:

Table VII shows the predicted versus measured gain for the 5 feeds.

Table VII Predicted versus measured gain

Feed Frequency GHz	Feed Type and position	Measured Gain, dB	Predicted Gain, dB	Error, dB
6.8	offset	40.80	39.38	1.42
10.7	offset 45	44.19	43.70	0.49
10.7	focus cp	44.38	44.20	0.18
10.7	offset VH	44.01	43.70	0.31
18.7	offset 45	48.46	48.00	0.46
18.7	focus cp	49.25	48.60	0.65
18.7	offset VH	48.68	48.00	0.68
23.8	offset	45.79	45.90	-0.11
37.	offset 45	53.03	53.40	-0.37
37.	focus cp	54.7	54.80	-0.10
37.	offset VH	53.56	53.40	0.16

IVd Range Error Budget:

Table VIII shows an error summary/allocation for the range measurement system.

Table VIII Polarization Purity Budget

Item	Error Contributor	Band Performance Value					Band Stokes Residual (dB)					Memo #
		37	23	18	10.7	6.8	37	23	18	10.7	6.8	
1	RF Range Contributions											
	* Ground Reflections(dB)	-70.9	-62.4	-65	-63.4	-56.7	-42.64	-38.48	-39.75	-38.97	-35.68	WL-20
2	* Guy Wire Reflections(dB)	-70.9	-62.4	-65	-63.4	-56.7	-42.64	-38.48	-39.75	-38.97	-35.68	WL-3/WL-20
3	**Transmit Antenna X-Pol(dB)	-81.2	-81.2	-78	-78	-65	-36.01	-36.01	-34.20	-34.20	-26.83	HB-1/WL-20
Subtotal RSS Range Multipath =							-35.82	-34.93	-33.89	-33.76	-26.76	
4	PRA Alignment Knowledge (deg)	0.019	0.019	0.019	0.019	0.019	-31.78	-31.78	-31.78	-31.78	-31.78	WL-7
Other Error Contributors												
5	Post Processing Errors	Truncation Analysis					-38.00	-38.00	-38.00	-38.00	-38.00	WL-8
6	Near Field Effects	0.36 Inch Near Field Focus					-38.00	-42.13	-44.26	-48.78	-52.71	WL-9
7	Instrumentation Phase Stability	3.02	2.55	1.51	0.84	0.76	-37.85	-38.01	-38.35	-38.57	-38.60	WL-10
8	Instrumentation Amp Stability	0.27	0.21	0.11	0.11	0.17	-38.40	-38.71	-39.23	-39.23	-38.92	WL-10
9	Tower/Reference Calibration Phase	5.19	0.33	4.28	1.00	2.00	-38.39	-39.99	-38.69	-39.77	-39.44	WL-11
10	Tower/Reference Calibration Amp	0.08	0.13	0.04	0.02	0.22	-42.48	-42.22	-42.69	-42.80	-41.76	WL-11
Subtotal RSS Others =							-34.56	-35.34	-35.40	-35.69	-35.54	
Total RSS Secondary Range =							-31.00	-31.01	-30.81	-30.81	-26.52	
Required=							-30.00	-23.00	-30.00	-30.00	-23.00	
Margin=							1.00	8.01	0.81	0.81	3.52	

* Average Over Bandwidth

** Max Value Since Doesn't Average. Value AT 37 GHz Based On Actual Theodolite Measurements Of Grid Conclusions Of HEB on 4-10-00

IVe Wide Angle Patterns and Beam Efficiency:

Wide angle pattern data was needed in order to predict antenna efficiency. The Melbourne range only allowed for pattern measurements out to ± 25 deg., due to multipath effects off the building structure at wider angles. The method used to determine efficiency was to validate the wide angle pattern predictions from the Ticra code using the range data out to ± 25 deg., and then use a composite of the range data and the Ticra code for antenna efficiency predictions. Predicted versus measured antenna patterns for the 18.7 VH feed at focus are presented in Fig. 28. Beam efficiencies are shown in Table IX. As can be seen, a reasonably good correlation is obtained.

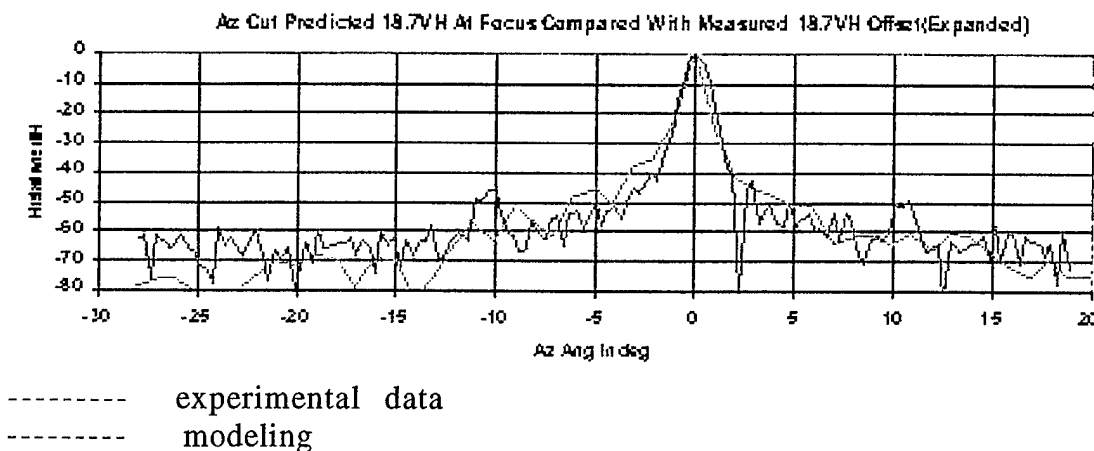


Fig. 28 Predicted versus measured antenna patterns, 18.7 VH focus feed

Table IX Beam Efficiencies from Composite of Range Data and Modeling

		Beam Efficiency For 3 Beamwidths Main Beam								
Band	3 BW's	PDR (1)	Predicted by Ticra (COI) (2)			Range + Predicted "Fill-in" (3)			Requir- ed	Pass /Fail(4)
	deg.		Az	El	Aver.	Az	EL	Aver.		
6.8	5.33		98.50	97.38	97.94	97.31	95.45	96.38	95.00	Pass
10.7	3.39	98.44	96.34	97.86	97.10	98.06	96.30	97.18	95.00	Pass
18.7	1.94	98.18	98.30	97.41	97.86	98.15	94.63	96.39	95.00	Pass
23.8	2.70(5)		95.29	96.95	96.12	97.37	95.36	96.37	95.00	Pass
37	0.98	97.08	99.56	97.83	98.69	97.26	93.15	95.21	95.00	Pass

Notes Referenced in Parenthesis:

1. PDR reported beam efficiency changed by COI 3 mil reflector from PDR estimated 5 mils.
 2. Predicted by Ticra includes manufactured COI contour for a focused VH feed principal planes.
 3. Range beam efficiency is calculated by substituting the measured regions range raw data for the equivalent predicted patterns regions from Ticra above.
 4. Final pass certification requires orbital life impacts (increased roughness etc.)
- Note 23.8 azimuth offset caused significant beam widening for the case. Pixel larger due to offset.

IVf Predicted versus Measured EIA and Scan:

Predicted versus initial measured EIA and Scan are presented in Tables X, XI and XII. The feed horn fixtures were designed with lateral and rotation adjustments to move the feeds into the correct placements. The requirement was for 0.018 deg. EIA between the feed triplets, and 0.5 deg. rotation alignment. Feed placement on the feed bench was initially determined by simulation with the Ticura Grasp8w code to meet these requirements. About half of the feeds were moved during the flight testing to meet these requirements.

Table X. EIA Values from original POE code (design values), and Ticura values with perfect parabola reflector

Feed Frequency GHz	Feed Type and position	EIA Values from original POE CODE Runs, deg.	EIA Values from TICRA CODE (perfect parabola) Runs, deg.	Range EIA Values , deg.	Difference between Ticura code and range, deg.
6.8	offset	53.54824	53.58586	53.5366	0.049
10.7	offset 45	49.94775	49.95417		
10.7	focus cp	49.94772	49.95417	49.9354	0.019
10.7	offset VH	49.94772	49.95417		
18.7	offset 45	55.3957	55.41090		
18.7	focus cp	55.3957	55.41090	55.3476	0.063
18.7	offset VH	55.3957	55.41090		
23.8	offset	53.04652	53.08668	53.00079	0.086
37.	offset 45	53.04652	53.08034		
37.	focus cp	53.04652	53.08034	52.953	0.13
37.	offset VH	53.04652	53.08034		

Table XI. Scan Angles Values from Ticura Runs and EIA Determination Program and Range Data

Feed Frequency GHz	Feed Type and position	Scan Values from Ticura Runs, deg.	Scan Values from Range data
6.8	offset	-11.066	-11.2853
10.7	offset 45	-5.2355	-5.46006
10.7	focus cp	-0.007446	-0.23114
10.7	offset VH	5.2206	5.006945
18.7	offset 45	-2.95685	-3.131566
18.7	focus cp	-0.003868	-0.204694
18.7	offset VH	2.9491	2.74159
23.8	offset	9.056	8.885
37.	offset 45	-2.53429	-2.7224
37.	focus cp	-0.0036926	-.2137
37.	offset VH	2.5269	2.294

Table XII. Total Scan Angles Values from Ticura Runs and EIA Determination Program and Range Data

Feed Frequency GHz	Total Scan Values between feeds from Ticura Runs, deg.	Scan Values between feeds from Range data, deg.	Difference in Total Scan Values (Ticura - range), deg.
10.7	10.456	10.467	-0.011
18.7	5.90595	5.873	0.03295
37.	5.0612	5.0164	0.0448

IVg Grid Spacing and Frequency Increments:

The pattern data was taken in a grid with pixels spaced ~ every 0.1 degree beamwidth. Data was taken over 21 frequencies across the horn bandwidth. The frequency data was taken so that the Stokes changes across the bandwidth could be averaged, particularly the changes caused by multipath reflections inside the horn caused by reflections off the Teflon feed covers. Table XIII shows the grid and frequency resolution for the 5 frequency bands.

Table XIII Grid Spacing and Frequency Increment

Freq. GHz	3dB Beam- width, deg.	Band- width, GHz	Pixel Size deg.	Freq. Incre- ment GHz	No. of Azimuth Points	No. of Elevation Points
6.8	1.9	0.125	0.174	0.00625	57	43
10.7	1.2	0.30	0.111	0.015	57	43
18.7	0.68	0.75	0.063	0.037	57	43
23.8	0.53	0.55	0.05	0.025	102	43
37.0	0.34	2.0	0.032	0.1	57	43

IVh Comparison of Range Patterns and Modeling:

Figs. 29-36 show comparisons of the range data contour patterns with the Grasp8w modeling. As can be seen, a good correlation was obtained.

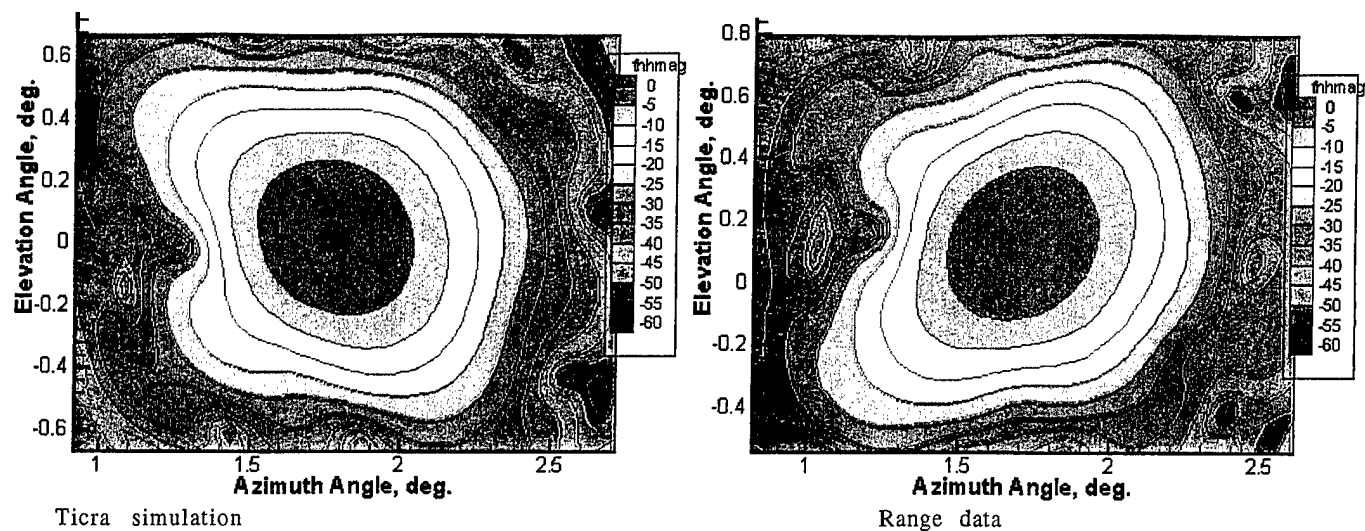


Fig. 29a & b HH magnitude, 37 GHz, VH offset feed

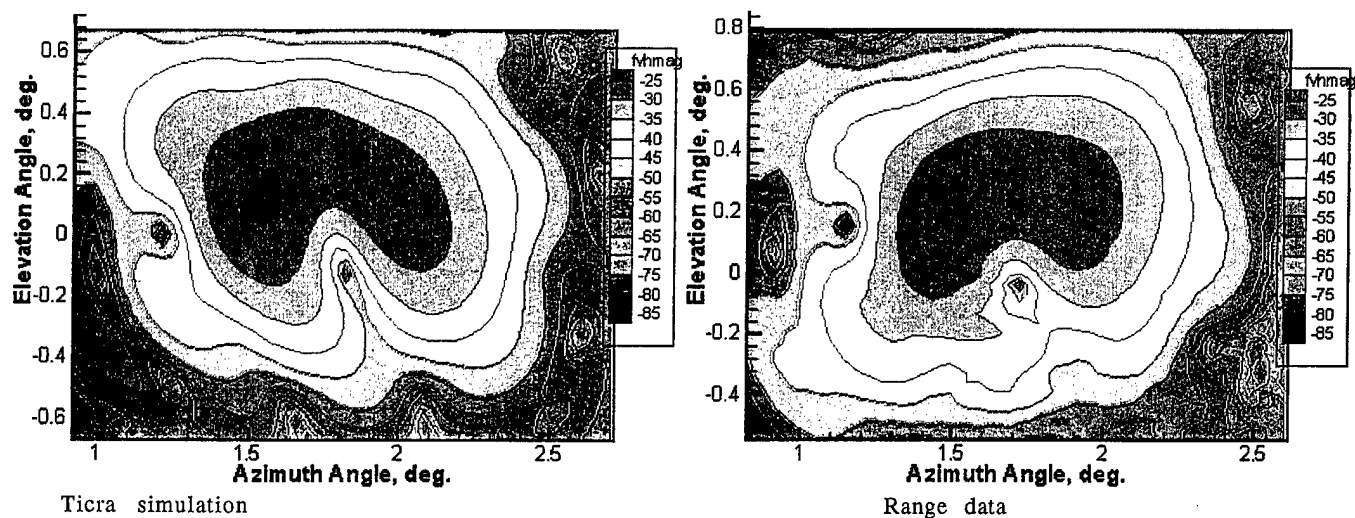


Fig. 30a & b VH magnitude, 37 GHz, VH offset feed

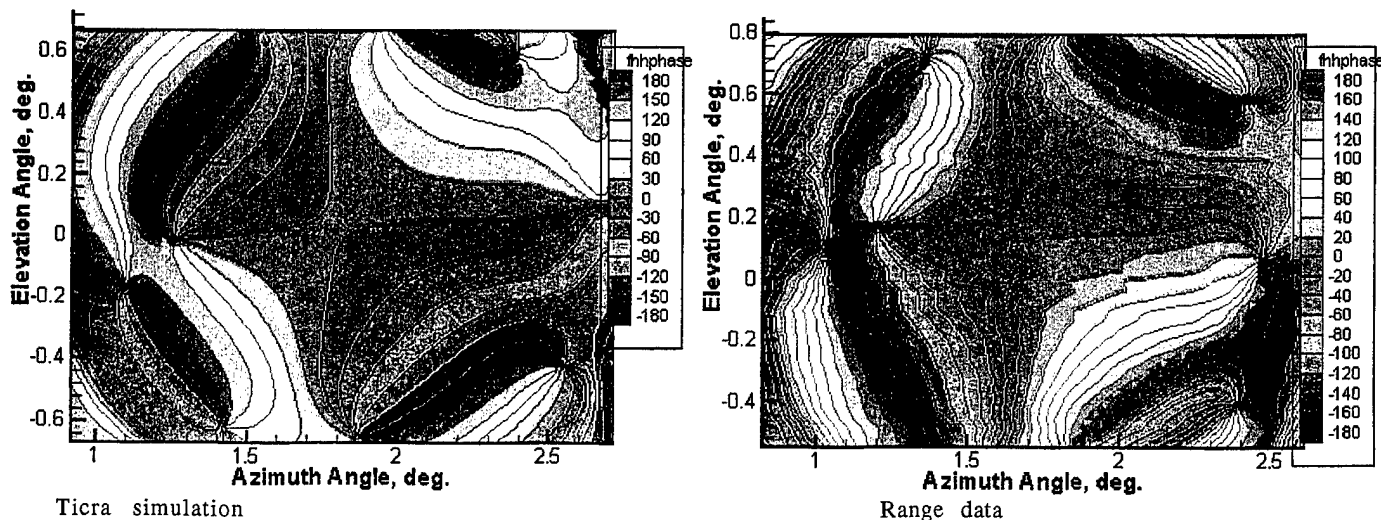


Fig. 31a & b HH phase, deg., 37 GHz, VH offset feed

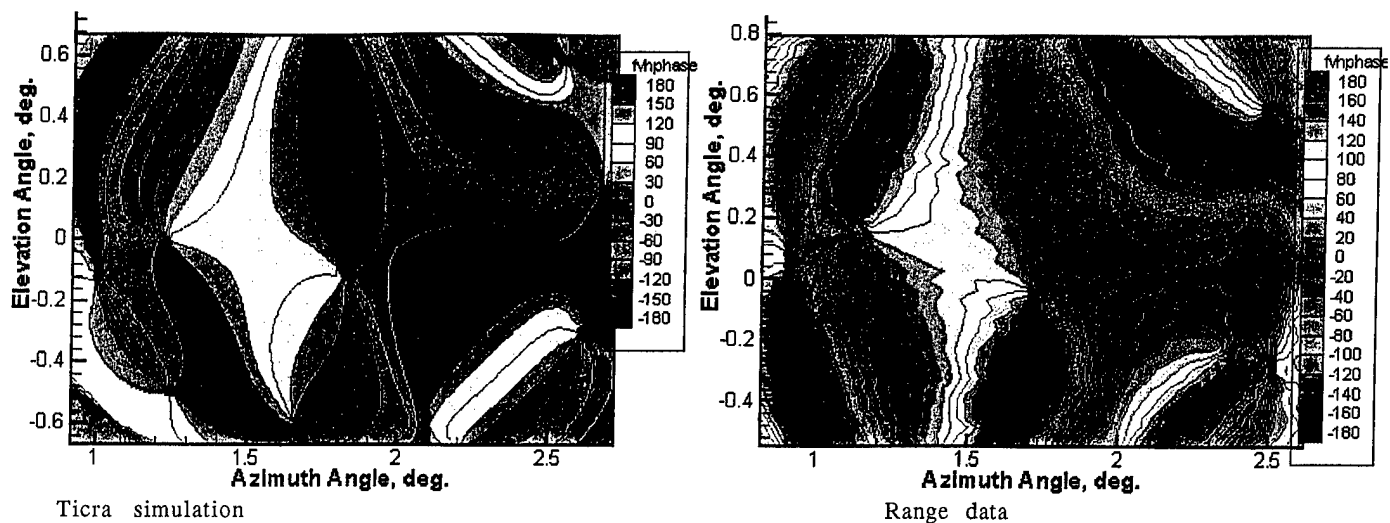


Fig. 32a & b VH phase, deg., 37 GHz, VH offset feed

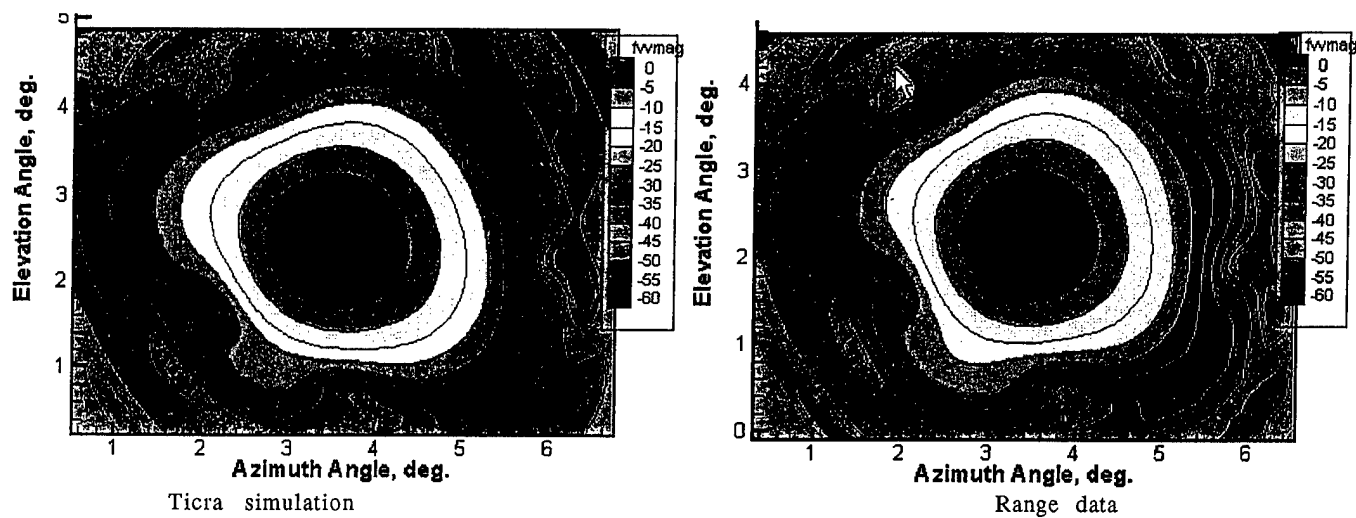


Fig. 33a & b VV magnitude, 10.7 GHz, VH offset feed

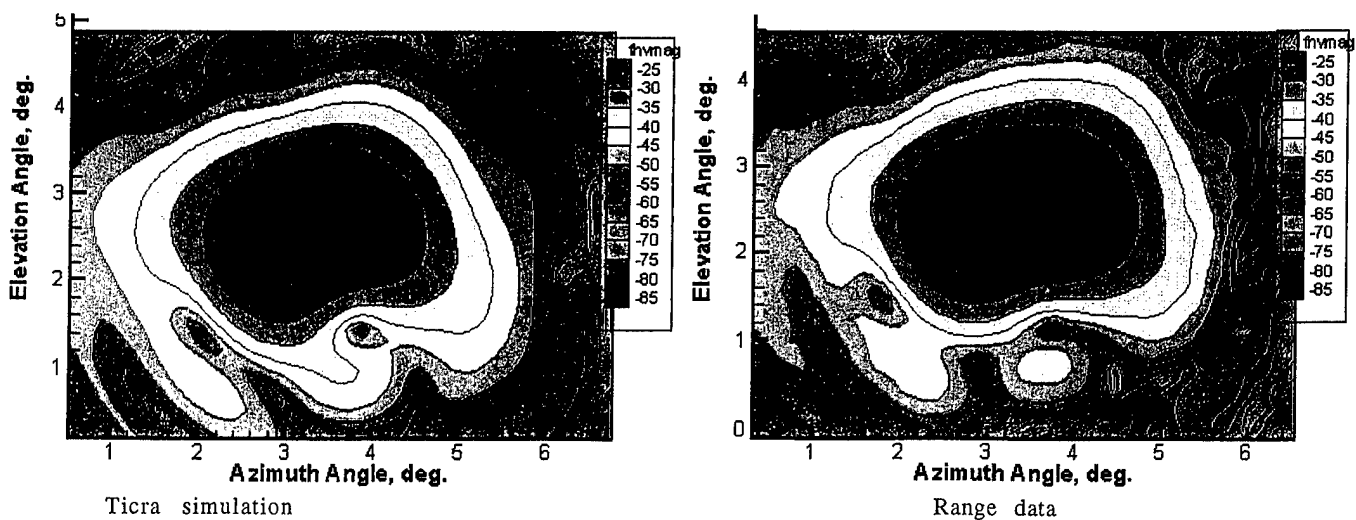


Fig. 34a & b HV magnitude, 10.7 GHz, VH offset feed

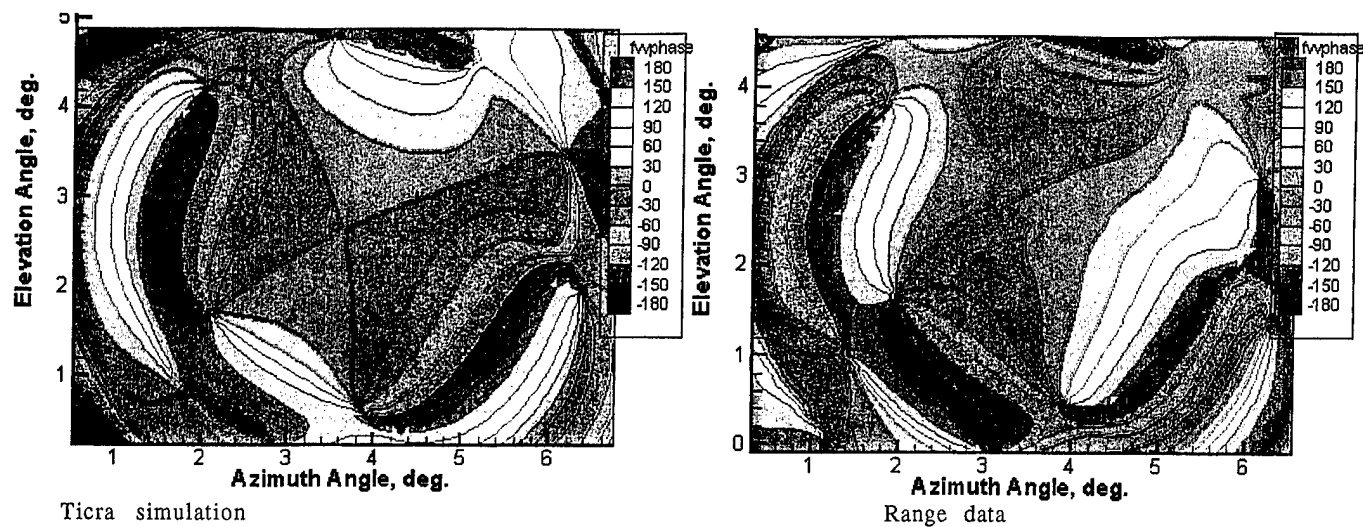


Fig. 35a & b VV phase, deg., 10.7 GHz, VH offset feed

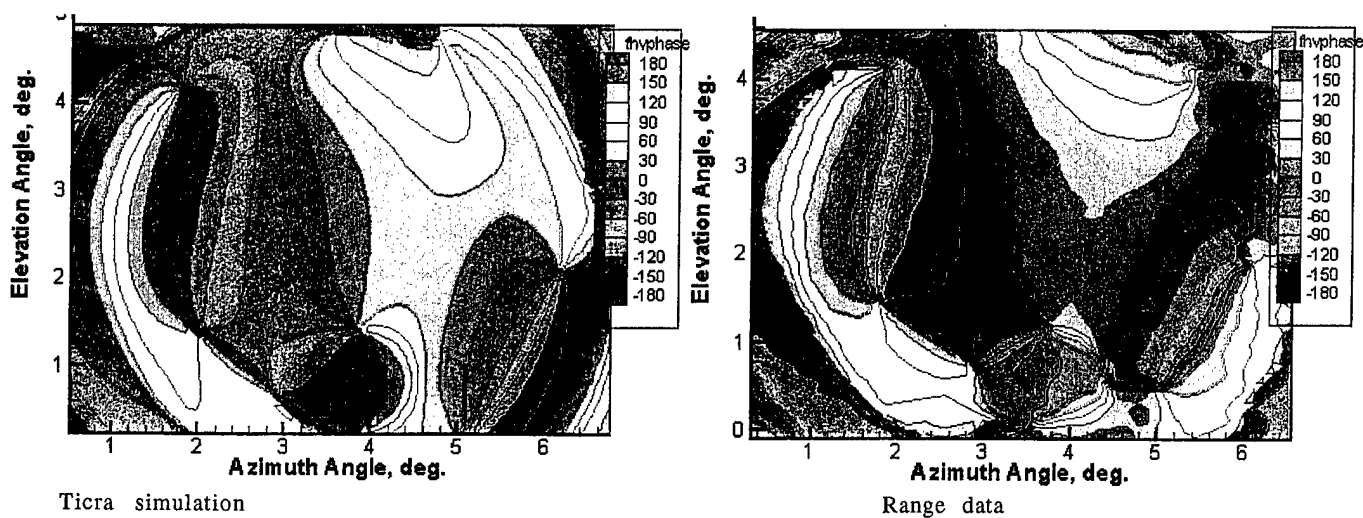


Fig. 36a & b HV phase, deg., 10.7 GHz, VH offset feed

IVi Stokes Matrix Results:

Figs. 37, 38, and 39 present the Stokes Matrix results for the 11 feeds. There were several critical features of the Stokes matrix that were checked to insure that the testing was in order. First of all, the matrix diagonal needed to be near 0 dB, which is equivalent to showing that most of the feed power was in the co-polarization. Given that the feeds were of high quality (VSWR and isolation checked with a network analyzer), the factors most influencing the matrix diagonal were the calibration factors used in the preprocessing of the data. The matrix diagonal was very sensitive to the calibration.

10.7vh					18.7vh				
	v	h	u	4		v	h	u	4
v	-0.014	-24.792	-41.839	-26.533	v	-0.018	-23.768	-25.809	-28.861
h	-24.630	-0.015	-31.879	-26.610	h	-23.643	-0.019	-41.418	-27.555
u	-28.934	-38.990	-0.030	-22.019	u	-28.028	-23.800	-0.043	-20.117
4	-23.423	-23.371	-22.077	-0.001	4	-25.088	-27.524	-19.878	-0.007
10.745					18.745				
	v	h	u	4		v	h	u	4
v	0.018	-24.698	-45.847	-29.537	v	-0.074	-23.724	-31.832	-22.298
h	-24.685	-0.048	-54.837	-28.126	h	-24.243	0.038	-41.946	-22.909
u	-46.601	-46.601	-0.030	-19.714	u	-44.206	-44.206	-0.043	-13.749
4	-24.972	-26.317	-19.790	-0.002	4	-20.226	-19.493	-13.644	-0.010
10.7cp					18.7cp				
	v	h	u	4		v	h	u	4
v	-0.002	-24.431	-24.686	-23.506	v	-0.011	-23.643	-24.717	-25.308
h	-24.586	-0.029	-26.874	-23.502	h	-23.913	-0.026	-19.359	-25.364
u	-23.964	-21.856	-0.032	-16.671	u	-16.395	-21.510	-0.038	-18.979
4	-20.416	-20.416	-16.729	-0.003	4	-22.820	-22.820	-19.318	-0.004
10.7Composite Of Above					18.7Composite Of Above				
	v	h	u	4		v	h	u	4
v	-0.014	-24.792	-41.839	-26.533	v	-0.018	-23.768	-25.809	-28.861
h	-24.630	-0.015	-31.879	-26.610	h	-23.643	-0.019	-41.418	-27.555
u	-46.601	-46.601	-0.030	-19.714	u	-44.206	-44.206	-0.043	-13.749
4	-20.416	-20.416	-16.729	-0.003	4	-22.820	-22.820	-19.318	-0.004

Fig. 37 Final Stokes Matrices, 10.7 and 18.7 GHz Polarametric Feeds

	37vh			
	v	h	u	4
v	-0.017	-24.152	-27.637	-22.046
h	-24.436	-0.016	-28.864	-21.374
u	-25.748	-24.397	-0.044	-19.538
4	-18.110	-18.731	-19.627	-0.013
	3745			
	v	h	u	4
v	-0.096	-23.854	-39.827	-27.731
h	-24.485	0.061	-36.753	-29.171
u	-45.723	-45.723	-0.038	-15.695
4	-24.602	-23.136	-15.803	-0.007
	37cp			
	v	h	u	4
v	-0.009	-22.956	-15.773	-18.159
h	-23.594	-0.032	-16.466	-18.148
u	-13.359	-12.744	-0.052	-18.046
4	-15.285	-15.285	-17.610	-0.017
	37Composite Of Above			
	v	h	u	4
v	-0.017	-24.152	-27.637	-22.046
h	-24.436	-0.016	-28.864	-21.374
u	-45.723	-45.723	-0.038	-15.695
4	-15.285	-15.285	-17.610	-0.017

Fig. 38 Final Stokes Matrices, 37 GHz Polarametric Feed

	23.8vh			
	v	h	u	4
v	-0.015	-24.616	-25.630	-28.282
h	-24.501	-0.015	-27.851	-24.213
u	-26.485	-23.009	-0.046	-15.166
4	-20.951	-24.761	-15.200	-0.018
	6.8vh			
	v	h	u	4
v	-0.019	-23.529	-24.109	-23.113
h	-23.594	-0.019	-23.221	-22.990
u	-20.874	-22.353	-0.095	-7.988
4	-19.381	-19.650	-8.005	-0.059

Fig. 39 Final Stokes Matrices, Non-Polarametric Feeds

Figs. 40-42 shows the variation of the 3rd Stokes over the frequency band for the 10.7, 18.7, and 37 GHz 45 flight feeds. As can be seen, the 3rd Stokes vary over the band; this variation is averaged out for the final Stokes matrix. From preliminary studies with the beta cloth cover on the feeds, it was apparent that the Stokes variations (then much larger) corresponded chiefly to the reflections from the front face of the feed. For the flight feeds, the small oscillations seen here are probably due to some range multipath effects.

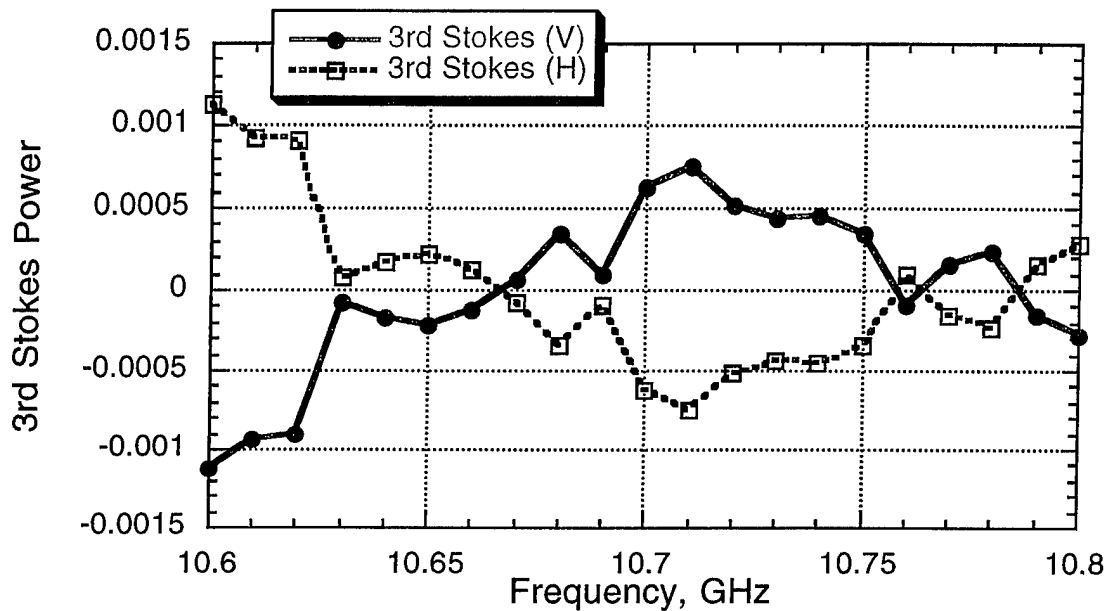


Fig. 40 10.7 GHz 45 feed (1/19/01), 3rd Stokes vs. Frequency

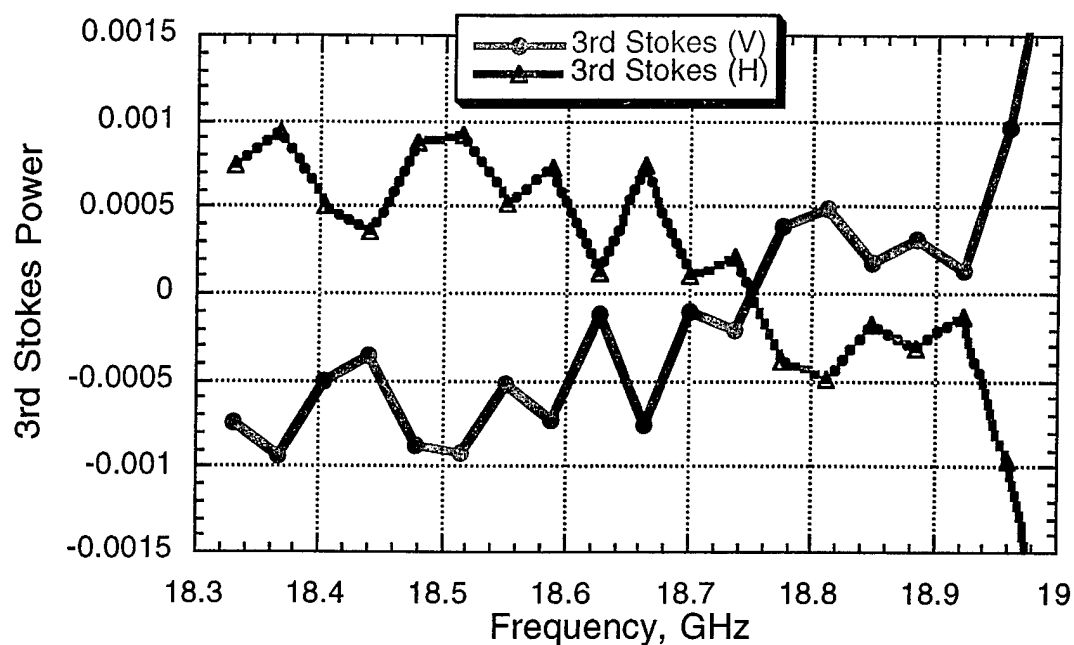


Fig. 41 18.7 GHz 45 feed (1/15/01), 3rd Stokes vs. Frequency

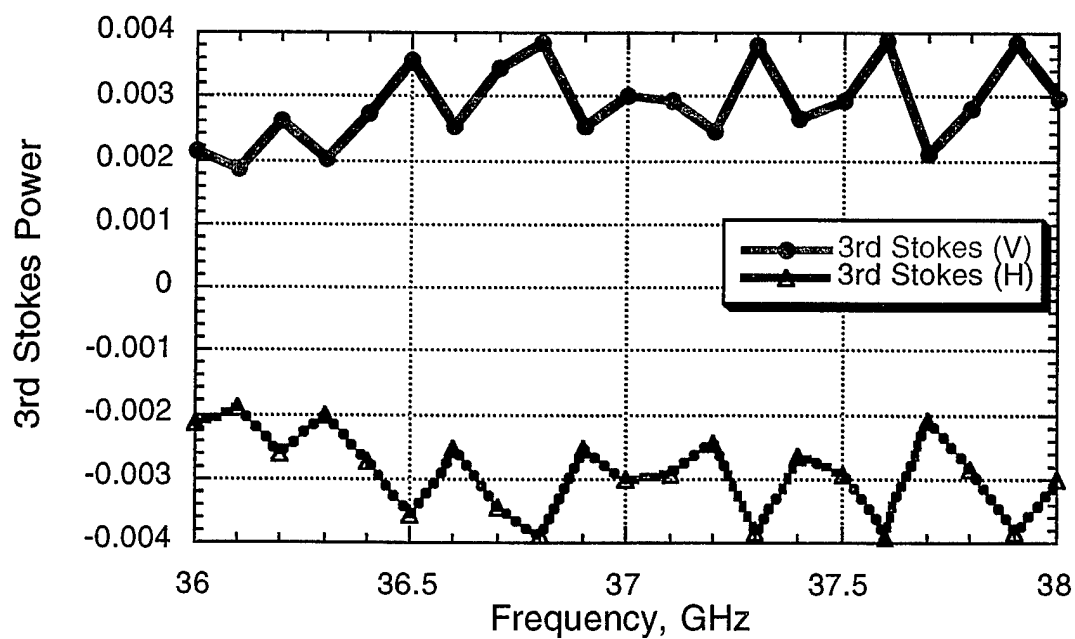


Fig. 42 37 GHz 45 feed (2/06/01), 3rd Stokes vs. Frequency, Case where 3rd Stokes are at -25.4 dB, not rotated to null out the 3rd Stokes' terms.

IVj Squint in the Patterns:

An unexpected characteristic of the measurements is that the cross-polarization peaks squint in elevation approximately 0.25 degree about the sum principal plane. The cross-pol for vertical tower polarization (Vx) squints in the negative elevation direction, whereas the cross-pol for the horizontal tower polarization (Hx) squints in the positive elevation direction. The feed cross-pol adding to the reflector cross-polarization produces this squint characteristic. Fig. 43 shows the squint amount relative to the feed crosspol level as modeled by the Grasp8w code.

Feed cross-pol measured on Florida range has -39 dB maximum cross-pol with ± 90 deg. phase quadrature. This used in Grasp8w model matches the magnitude squint seen in range data. If the feed is modeled with 0 and 180 deg. phase quadrature, the cross-pol magnitude squint is replaced by squint in the phase of the cross-pol.

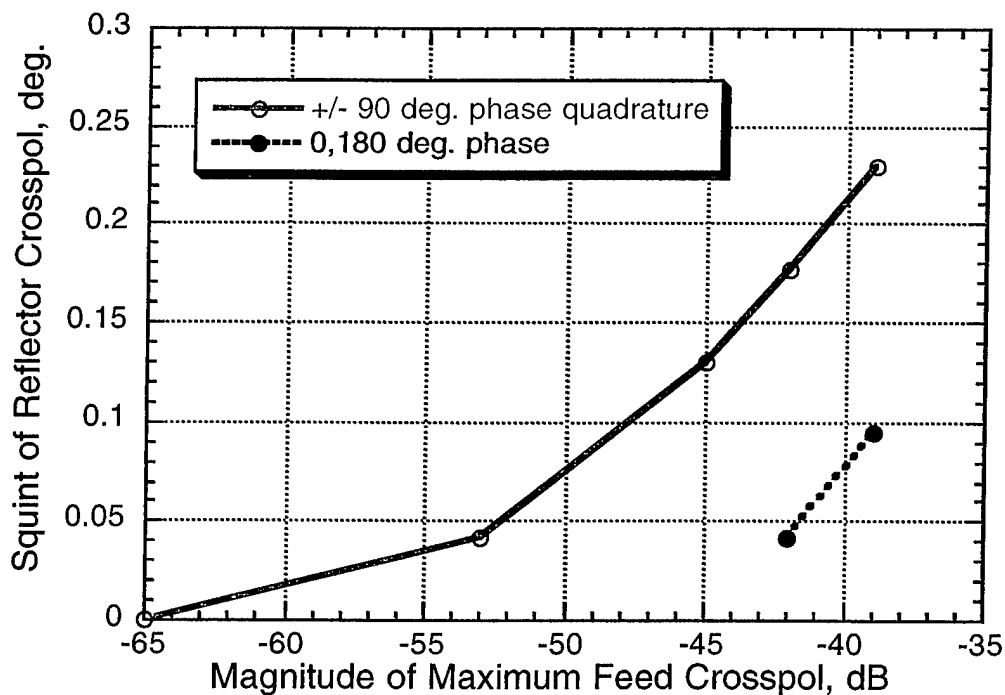


Fig. 43 Squint amount versus feed cross-pol level as modeled by the Grasp8w code

V. COLD and WARM LOADS

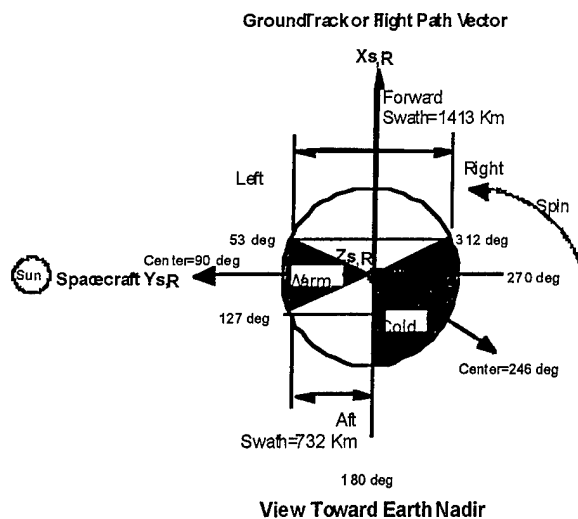
Warm and cold loads are used to calibrate the receivers. They provide known brightness temperature inputs for the on-orbit calibration. The blackbody warm load is a cast aluminum plate with many pyramids facing the feeds. This face is coated with Eccosorb CR-117 Epoxy coating. The temperature of the warm load is measured to an accuracy of ± 0.05 °C continuously. The shape was designed to minimize blockage. The cold load is a small offset reflector that is used to position the beam into cold space. These loads are fixed in place and the reflector/feed system is spun around them to provide calibration data once per scan. Fig. 44 shows the cold and warm loads as well as the approximate swaths they have in the scan. Figs. 45 and 46 show the cold load reflector configuration

The range was also used to determine the forward and aft swaths of the system. The Stokes matrix error allocation of interference from the warm and cold loads in the feed vicinity was determined to be -36 dB. The initial positions of the cold and warm load to cause this amount of interference was determined by modeling the cold and warm loads with the Ticsra Grasp8w code. The cold load was modeled accurately as an offset parabola, the warm load was approximated as a flat plate. Fig. 45a shows the cold load in the Grasp8w modeling. Fig. 44a shows a photo of the warm load in an obscuration testing position. Fig. 47 shows a correlation of the Ticsra modeling and range results for the 10.7 GHz 45 feed. In general, there was good agreement with the modeling and the range data. The Ticsra predicted closely the interference with the loads and the reflector beam. It underestimated the interference due to feed blockage. This is undoubtedly due to the inaccuracy of the estimates of the code for objects in the near-field.

The modeling and range data were used to analyze how the warm and cold load blockage impacts the Stokes parameters as the loads scan across the feeds. The exact obscuration of the warm and cold load swaths will be determined on orbit. This data can be easily obtain as the satellite passes by a coastline where the radiation data changes rapidly. The range testing and modeling was used to determine the quality of the feed obscuration. It was hoped that the obscuration was in the form of a steep dropoff in accuracy of the Stokes. This was confirmed by the testing and modeling.

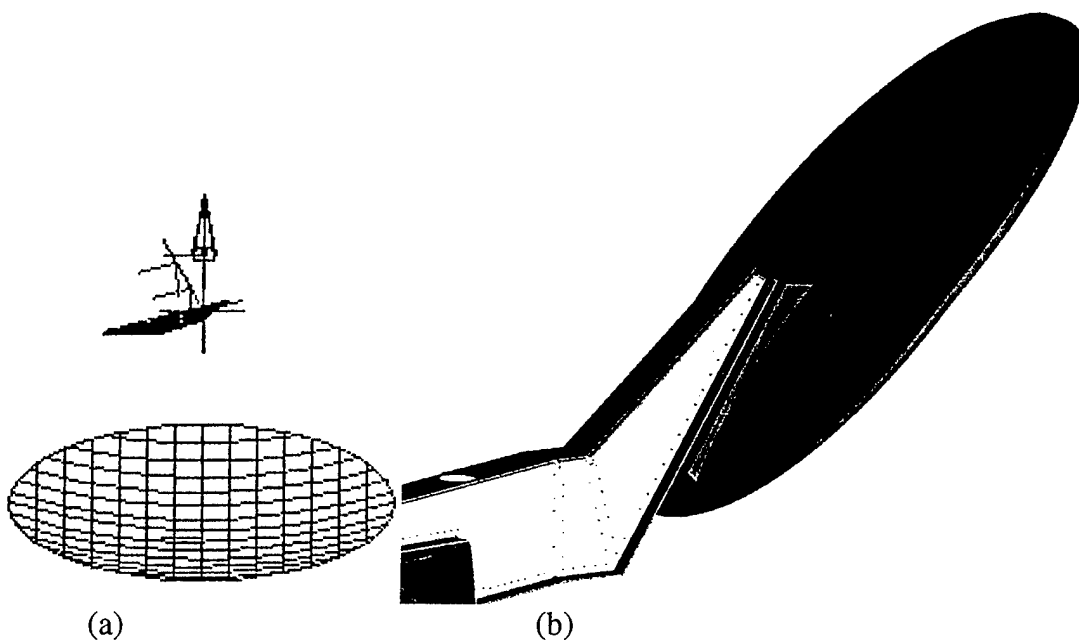


(a)



(b)

Fig. 44 (a) Range obscuration testing with warm load partially obscuring the 6.8 Ghz feed (b) Ground track on earth showing approximate obscuration zones from warm and cold load.



(a)

(b)

Fig. 45 Cold load configuration (a) modeled with Grasp8w code (b) CAD model

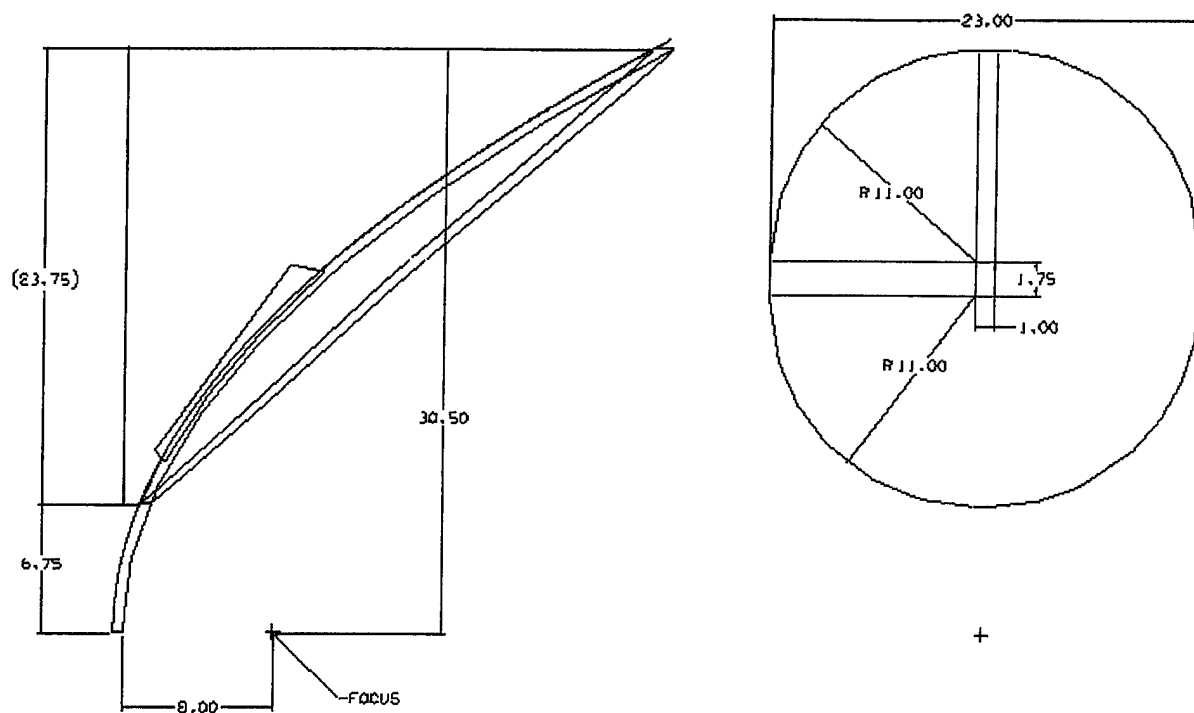


Fig. 46 Cold load configuration

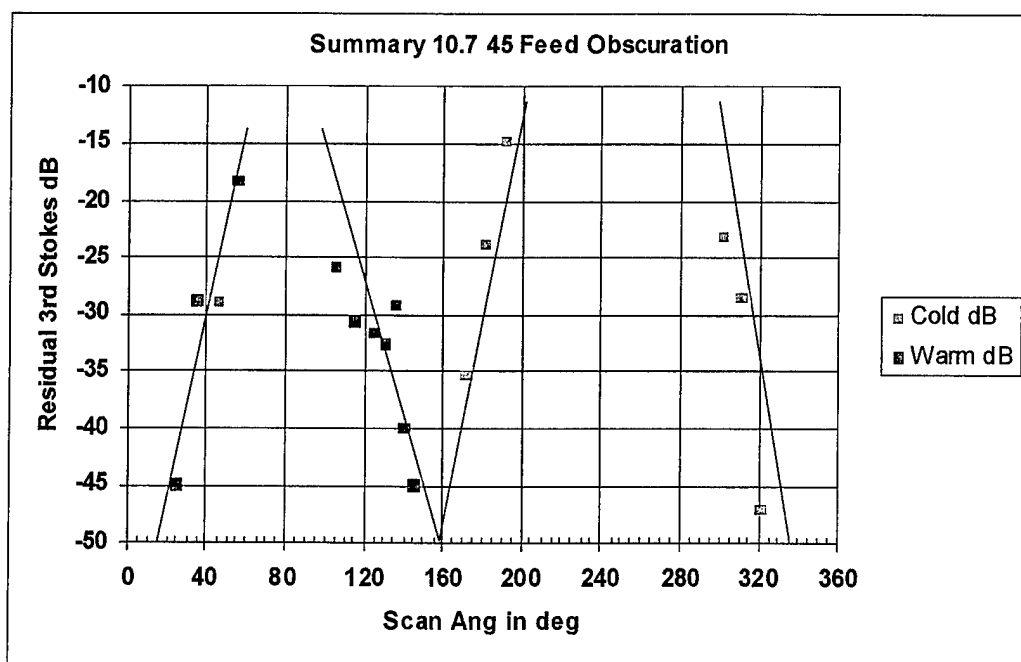


Fig. 47 Range data and modeling (range data is shown in points, modeling is represented by the black lines)

VI. COMPUTER CODES FOR STOKES VECTOR CALIBRATION

VIa General Comments:

The Stokes coupling matrix represents the inherent coupling of four polarizations into the vertical, horizontal, ± 45 degree, and RCP/LCP (right and left hand circular polarization). To generate that matrix, range data is taken over a ± 1.5 beamwidth patterns of co- and cross- polarization amplitude and phase. This data is then input into the POE code, which transforms the patterns into earth-based polarizations which can be integrated over the entire beam to solve for the Stokes matrix. For completeness, the POE code for the ± 45 degree polarization is listed in Appendix D. The computer listing of a preprocessor code which takes the raw range data and prepares it for input to the POE code (including all range calibration factors) is not listed.

The POE code is actually made up of three separate codes, a V/H code, a ± 45 degree code, and a RCP/LCP code. Each code analyzes data from the respective polarization horn data to create the Stokes matrix terms for those polarizations. The codes will each form an entire Stokes matrix, but only the terms of the matrix for the input polarization are accurate, the other terms are approximations. The V/H and RCP/LCP codes are not listed in their entirety, but pertinent excerpts that represent the major differences between those codes and the ± 45 degree POE code are given in Appendix E.

For the ± 45 degree polarizations and V/H polarizations, the code can iteratively determine the angle of rotation needed to align the polarization with the earth polarization. For offset feeds, this angle is called the toe-in angle. The polarization axis must be aligned at an angle different from the spacecraft vertical and horizontal because the beam hits the earth at an angle and must be aligned with the earth 'horizon'. Calculated toe-in angles for the 11 feeds were given in Table III along with the feed positions on the feed bench and the azimuth and elevation pointing angles of the beams. The earth incidence angle for the focus feed is 53.04652 deg. X-displacement affects elevation, a positive x position makes a downward (negative elevation), y-displacement affects azimuth, a positive y position moves the beam (looking outward from the reflector) to the right (positive) direction.

VIb Range Preprocessing of Data and Calibration:

The preprocessing of the data included several steps. First off, various timing constants relative to the data collection system are applied. The data is separated out into the various frequencies, and the elevation phase correction added. The elevation phase correction is to correct for the nodding phase for visualization purposes only, it has no impact on the Stokes matrix calculations. The preprocessor also has the ability to do a data interpolation (to evenly space

the data for the integration in the post-processing codes), but this was not needed in the final flight data collection, since the data spacing was adequate for the post processing. The centroid angles of the data are also found.

Next the range calibration is done. This is necessary for calibrating the reference antenna and subsequently the cables/waveguide to the feed ports. The calibration is done in two steps. First, the reference antenna amplitude and phase cal offsets are applied to all the data coming from the horizontal tower polarization. These calibration factors (21 frequencies per band) are determined by sampling the two orthogonal components incoming from each source antenna when rotated 0 to 90 degrees. This sampling is performed in the 23.8 and 6.8 GHz bands by rotating the Windsat feeds in synchronization with the source antenna rotation from vertical to horizontal polarization and logging the coherent data with the network analyzer data acquisition system. In the 37, 18.7, and 10.7 Bands where CP feeds were available, enabling no feed configuration disturbance by rotating, the sampling of the source vertical and horizontal polarizations is obtained without the need for rotating the feeds. The impact of the feeds' CP axial ratio is well within the error budget to determine the calibration. Next the co-polarization phases at boresight are computed (45 deg. data needed to be first electromagnetically rotated to V/H position to find these) and differenced; this difference is applied to all the data from one of the ports (co- and cross-pol) so that the boresight phases from both co-pols is the same (port phase calibration).

Electromagnetic rotations of the fields are done in various sections of the code: for applying the calibration factors properly, to null out the 3rd Stokes (to determine the feed rotational alignment), and to allow the data to be used for the various post-processing codes (which require V/H or 45 data as input). Care is taken that the field rotations are done properly, either between the fields from each port separately ('port' rotation), or the fields from each tower separately (tower rotation), depending on what calibrations have been applied with the rotation is done. The idea is to minimize the impact of calibration on the rotation accuracy. Because the 'port' calibration is done after the 'tower' calibration, it is necessarily less accurate, so a tower rotation is always preferable. When the rotation is done to determine the co- and cross-pols during the 45 deg. data cal, however, the rotation occurs on each port initially, then switches back to a tower rotation when the port calibration phase has been determined. If there was no error in the calibration factors, the port and tower rotations would give identical results. Fig. 48 shows the FORTRAN coding for the separate electromagnetic rotations on the complex variables.


```

C      cavcomag(j,k)= vertical polarization from tower, vertical port
C      cavcrossmag(j,k)= vertical polarization from tower, horizontal port
C      cahcomag(j,k)= horizontal polarization from tower, horizontal port
C      cahcrossmag(j,k)= horizontal polarization from tower, vertical port
      ca1=cavcomag(j,k)
      ca2=cavcrossmag(j,k)
      ca3=cahcomag(j,k)
      ca4=cahcrossmag(j,k)

C      this rotation technique rotates the ports, not the tower
      cavcomag(j,k)=COSPT*ca1+SINPT*ca4
      cahcrossmag(j,k)=-SINPT*ca1+COSPT*ca4
      cavcrossmag(j,k)=COSPT*ca2+SINPT*ca3
C      cahcomag(j,k)=-SINPT*ca2+COSPT*ca3

C      this rotation technique rotates the towers, not the ports
      cavcomag(j,k)=COSPT*ca1-SINPT*ca2
      cavcrossmag(j,k)=SINPT*ca1+COSPT*ca2
      cahcrossmag(j,k)=COSPT*ca4-SINPT*ca3
      cahcomag(j,k)=SINPT*ca4+COSPT*ca3

```

Fig. 48 FORTRAN coding for the separate electromagnetic rotations on the complex variables

The total power in each port is computed next and made equal (this is the port magnitude calibration), and a complex conjugate is applied because we want to switch from receiving to transmitting the powers. The data is then written out for the post-processing code as well as for a 2D plotting program.

The range calibration file used for the flight 37GHz feeds are shown in Table XIV.

Table XIV Range Calibration File (37 GHz feeds)

Freq. GHz	Mag. cal dB	Phase Cal deg.	Freq. GHz	Mag. cal dB	Phase Cal deg.
36.0	-0.07	3.13	37.1	-0.02	2.49
36.1	-0.10	3.12	37.2	0.05	2.49
36.2	-0.11	3.11	37.3	0.12	1.98
36.3	-0.11	3.13	37.4	0.05	1.65
36.4	-0.05	2.98	37.5	-0.03	1.56
36.5	-0.04	2.83	37.6	-0.08	1.44
36.6	-0.04	2.65	37.7	-0.13	1.63
36.7	-0.09	2.37	37.8	-0.15	2.40
36.8	-0.11	2.88	37.9	-0.09	3.22
36.9	-0.01	2.94	38.0	0.00	3.22
37.0	0.01	2.54			

VIc Range Stokes Processing:

The structure of the POE code starts by centroiding the copol patterns to define the boresight. The beam energies are then normalized. At this point any offsets from the azimuth and elevation angles on the range with respect to the spacecraft coordinate system are applied through coordinate transformations. The polarization vectors are then 'mapped' into an earth-based system (to take into account that the polarization sampled off the earth will be rotated with respect to the spacecraft vertical/horizontal for any offset angles), and the various polarization components are summed to form the Stoke's matrix.

The Stoke's matrix terms for each frequency are then averaged (along with receiver bandwidth weighting) to form the final matrix that covers the whole feed band.

One of the confusing points in developing the Stokes processing codes was setting up the conversion from the azimuth and elevation angles from the range into the direction cosines for the POE code. This also affects weighting of the angles in the centroid calculations. The conversion is dependent on the range positioner being either azimuth over elevation (az over el) or elevation over azimuth (el over az). The elevation over azimuth coordinate transformation is the more intuitive, since the angles are very similar to a theta/phi conversion. However, our range had the azimuth positioner over the elevation positioner, so we needed to use the az over el coordinate transformation. Basically, the conversion matrix for the az/el to direction cosines is found by multiplying the azimuth rotation matrix by the elevation rotation matrix ($[el]*[az]$). (Note: for el over az the formula is $[az]*[el]$).

For the ± 45 degree polarizations and V/H polarizations, there is a standard exponential curve (loosely named the 'bat' curve) that maps the 3rd Stokes terms into the feed angle of rotation. The null in this curve corresponds to where the polarizations of the feed are aligned to the earth 'horizon'. During flight testing and configuration, the feed is rotated so that the null is within ± 0.5 deg., as determined by the 'bat' curve. During testing, the null angle as determined by the 'bat' curve was verified to track with mechanical feed rotation, and to be an accurate gauge of the mechanical feed position. Fig. 49 shows the 'bat' curve for a 10.7 GHz 45. deg. polarization offset feed (during prototype testing).

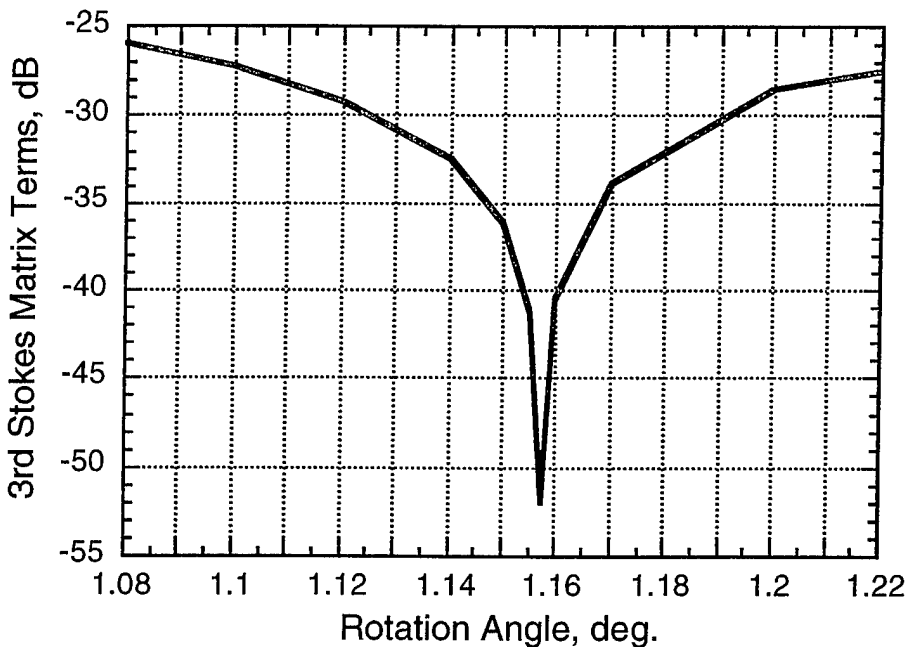


Fig. 49 'Bat' curve for 45 offset feed with toe-in misalignment of 1.157 deg.

VII. SUMMARY

An overview of the Windsat antenna design, modeling and testing was presented. In general, the reflector modeling codes proved invaluable for modeling the system performance and matched the range measurements well. A unique range capability was developed capable of the tight measurement requirements for the remote sensing system.

Acknowledgments

Many others were involved with the successful completion of the Windsat antenna development and ground testing. Gene Poe developed the equations for predicting the Stokes matrices from the range data. Stefan Cottle did the feed bench design. Jeff Benson developed all the data acquisition software and designed the tower instrumentation. Jeff Gutwein aided with the range measurements. Steve White designed and installed all the mechanical subsystems on the range. Dave Spencer was the Windsat Project Manager.

REFERENCES

- [1] Chang, P.S., P.W. Gaiser, L. Li, and K.M. St. Germain, "Multi-frequency polarimetric microwave ocean wind direction retrievals", Proceedings of the International Geoscience and Remote Sensing Symposium, 1997, Singapore, 1997.
- [2] Fawwaz T. Ulaby, Richard K. Moore, Adrian K. Fung, *Microwave Remote Sensing: Active and Passive, Vol. I, Microwave Remote Sensing Fundamentals and Radiometry*, ISBN 0-201-10759-7, Addison-Wesley Publishing Company, Inc.
- [3] N. Skou, B. Laursen, and S. Sobjerg, "Polarimetric radiometer configurations: Potential accuracy and sensitivity," IEEE Trans. on Geoscience and Remote Sensing, Vol. 37, pp. 2165-2171, Sept. 1999
- [4] B. Laursen, N. Skou, "Wind Direction over the Ocean Determined by an Airborne, Imaging, Polarimetric Radiometer System", IEEE Trans. on Geoscience and Remote Sensing, Vol. 39, No. 7, pp. 1547-1555, July 2001.
- [5] S. H. Yueh, W.J. Wilson, S.V.Nghium, F.K. Li, W. B. Ricketts, "Polarimetric measurements of sea surface brightness temperatures using an aircraft K-band radiometer," IEEE Trans. Geosci. Remote Sensing, vol. 33, pp. 85-92, Jan. 1995.
- [6] S. H. Yueh, W.J. Wilson, S.V.Nghium, F.K. Li, W. B. Ricketts, "Polarimetric brightness temperatures of sea surfaces measured with aircraft K- and Ka-band radiometers," IEEE Trans. Geosci. Remote Sensing, vol. 35, pp. 1177-1187, Sept. 1997.
- [7] N. Skou and B. Lauersen, "Measurement of ocean wind vector by an airborne, imaging polarimetric radiometer," Radio Sci., vol. 33, pp. 669-675, May-June 1998.
- [8] S.H. Yueh, W. J. Wilson, S.J. Dinardo, and F.K. Li, "Polarimetric microwave brightness signatures of ocean wind directions," IEEE Tran. Geosci. Remote Sensing, vol. 37, pp 949-959, Mar. 1999.
- [9] J.R. Piepmeier, A.J. Gasiewski, M.Klein, V. Boehm, and R.C. Lum, "Ocean surface wind direction measurement by scanning polarimetric microwave radiometry," in Proc. Int. Geoscience and Remote Sensing Symp. '98, Seattle, WA, p. 2307.
- [10] A.J. Gasiewski and D.B. Kunkee, "Calibration and application of polarization-correlating radiometers," IEEE Trans. Microwave Theory Tech., vol. 41, pp. 767-773, May 1993.
- [11] John Ruze, "Antenna Tolerance Theory - A Review", Proc. IEEE , Vol. 54, pp. 633-640, April 1966.
- [12] W. L. Lippincott, "Antenna Modeling and Data Analysis Reports, RT Memo Set", Naval Research Laboratory, Washington, D.C., 2000-2001.
- [13] T. Gutwein, "Antenna Range Daily Test Logs and Data Analysis Reports Including Final Acceptance Testing Compliance Document", Microstar Inc., Melbourne, Florida, 2000-2001.

Appendix A.

Radiometer Performance Overview

The Stokes coupling matrix represents the cross coupling of four polarizations into the vertical, horizontal, +/- 45 degree, and RCP/LCP (right and left circular polarization) (Fig. A1). The Stokes matrix represents the calibrated performance of the antenna, and does not change on-orbit. The antenna must be designed and calibrated carefully to insure the Stokes matrix accuracy to within a 30 dB residual error. This report addresses the design and calibration of the antenna.

The inverted Stokes matrix is multiplied by the antenna temperatures (powers) obtained on-orbit to determine the scene temperature. The scene temperature varies with wind speed and direction in a sinusoidal fashion (Fig. A2) for both the 3rd and 4th Stokes terms. Algorithms are used to determine wind speed and direction from the scene temperature, taking into account the data for both the 3rd and 4th Stokes terms at the different frequencies. Small errors in the 3rd and 4th Stokes terms create large errors in the scene temperatures.

$$M = \begin{bmatrix} C_{vv} & C_{vh} & C_{vu} & C_{v4} \\ C_{hv} & C_{hh} & C_{hu} & C_{h4} \\ C_{uv} & C_{uh} & C_{uu} & C_{u4} \\ C_{4v} & C_{4h} & C_{4u} & C_{44} \end{bmatrix}$$

Fig. A1 Stokes Coupling Matrix; v represents vertical polarization, h is horizontal, u is +/- 45 deg., and 4 is RCP/LCP. Term C_{uv} is the coupling of vertical polarization into the +/- 45 deg. polarization horn.

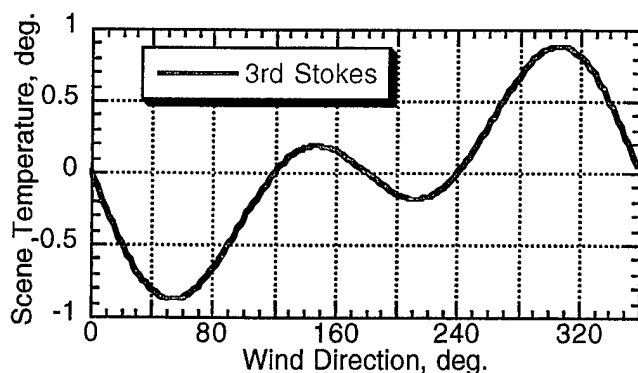


Fig. A2 Sample Scene Temperature vs. Wind Direction Plot for the 3rd Stokes at one frequency

Ocean surface emission varies with the wind vector (speed and direction). This has been validated by aircraft measurements. The wind direction dependence arises from anisotropic distribution and orientation of wind driven waves. The Stokes Vector describes the polarization properties of emitted radiation in terms of the "brightness temperature", T_B . Figs. A3, A4, and A5 present equations and diagrams to illustrate this.

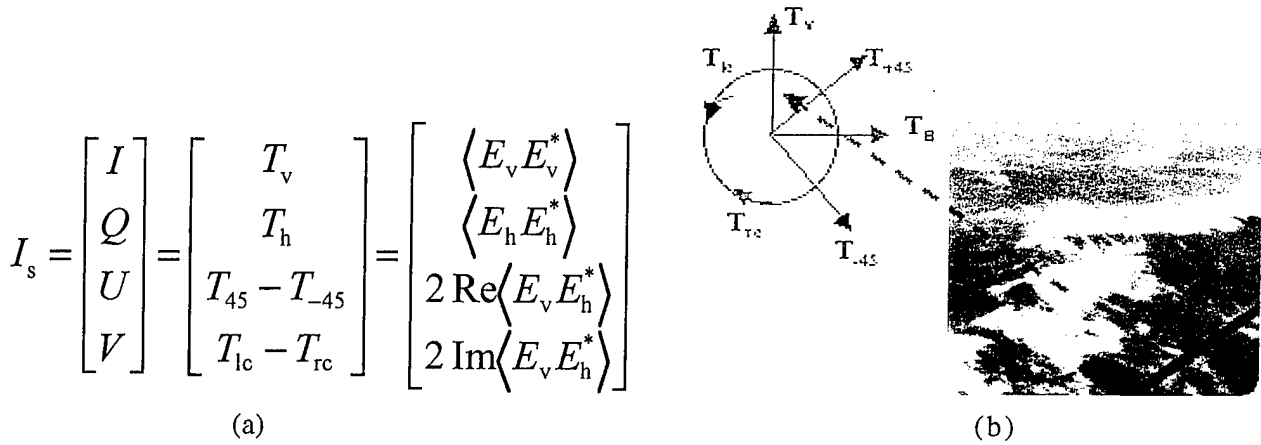
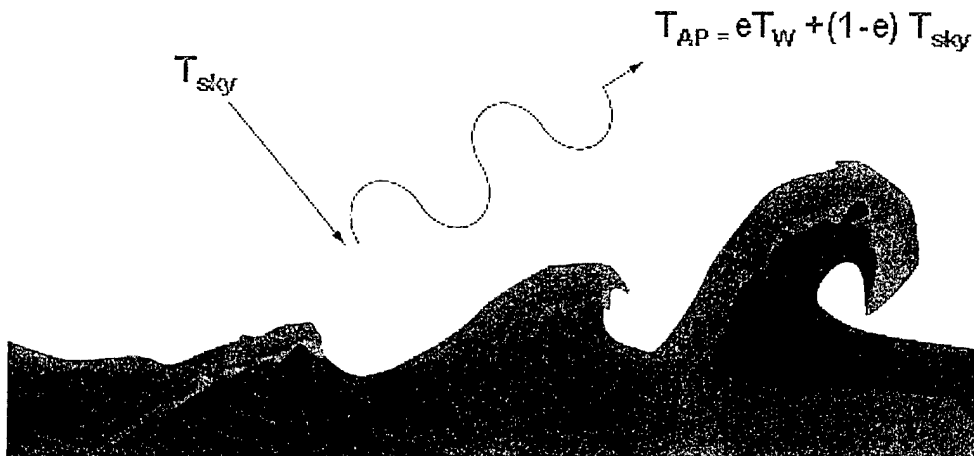


Fig. A3 (a) Stokes Vector (I_s) in terms of vertical (v) and horizontal (h) electric field vectors (E), and brightness temperatures (T). 45 and -45 refer to orientation of polarization, lc and rc represent left and right hand polarization. I , Q , U , and V are the 1st, 2nd, 3rd, and 4th Stokes terms. (b) a depiction of radiation components of vertical (V), horizontal (H), 45 deg. polarization and circular polarization

$$I_{scene} = [M_s]^{-1} * I_{antenna}$$

Fig. A4 Equation of Stokes vector (I_s) derived from multiplication of inverted Stokes matrix (M_s) with received radiation (I_a)



$$T_B(p, \nu, \theta, \varphi) = L\{(1-F)[eT_w + (1-e)T_{sky}]_{sea} + F[eT_w + (1-e)T_{sky}]_{foam}\} + T_{atm}$$

$$e(p, \nu, \theta, \varphi) = e_{spec} + [\Delta e_0 + \Delta e_1 \cos \varphi + \Delta e_2 \cos 2\varphi]_{rough}$$

e = emissivity

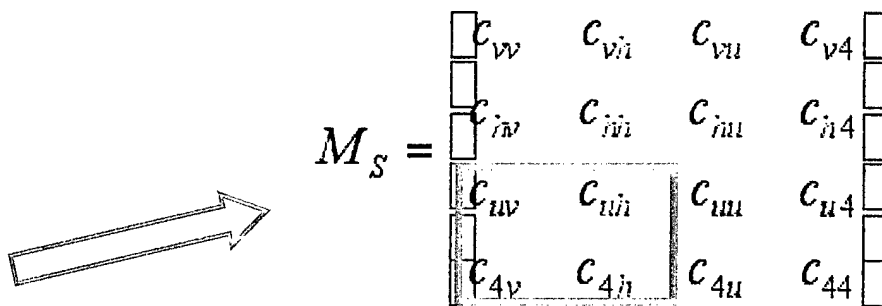
L = atmospheric loss

F = percent of foam coverage

Fig. A5 Brightness temperature resulting from wave roughness induced by wind

Antenna cross-polarization characterization is a critical part of antenna pattern correction for a polarimetric radiometer. This correction is applied through multiplication of the received antenna polarization components with the Stokes matrix. The Stokes matrix represents the inherent cross-polarization components of the antenna. Basically, if the antenna was perfect, and had no cross-polarization components, then the Stokes matrix would be the identity matrix with the diagonal components equal to 1 and all other components equal to 0. Since this is never the case, the antenna cross-polarization components are characterized and manipulated mathematically into the Stokes matrix form. The Stokes coupling matrix represents the coupling of Stokes parameters into one another.

Small errors in the 3rd and 4th Stokes coupling terms (Fig. A6) create large errors in scene temperature. Therefore, these parameters are the most critical in the calibration process.



$$M_S = \begin{bmatrix} c_{vv} & c_{vh} & c_{vu} & c_{v4} \\ c_{hv} & c_{hh} & c_{hu} & c_{h4} \\ c_{uv} & c_{uh} & c_{uu} & c_{u4} \\ c_{4v} & c_{4h} & c_{4u} & c_{44} \end{bmatrix}$$

Fig. A6 Stokes matrix (M_S) with the important 3rd and 4th Stokes coupling terms highlighted. c_{hv} is the coupling of V (vertical) into H (horizontal) polarization

Rotation of the antenna polarization basis relative to the surface of the earth introduces additional cross-polarization errors. Changes in the earth incidence angle (EIA) changes the brightness temperature measurement. Small changes in these angles are tolerable given sufficient knowledge of the true angles, however, knowledge errors produce performance errors. Fig. A7 defines the various pointing and polarization angles relative to the spacecraft and earth used in the Windsat system.

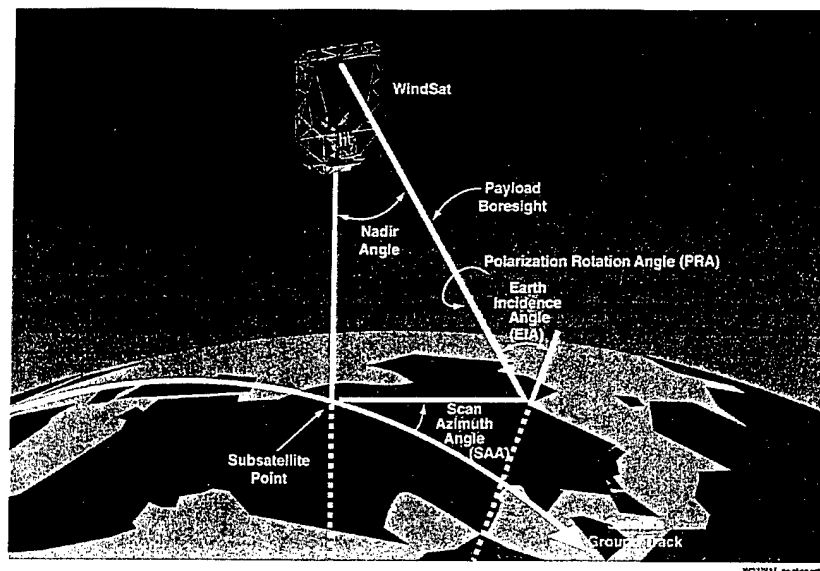


Fig. A7 Spacecraft and earth angle definitions

Calibration error consists of the weighted combination of several error sources: calibration target uncertainty, non-linearity, gain drift, quantization errors, and antenna beam efficiency. Because T_u and T_4 are derived from the difference of two principal polarizations, some errors will cancel since the channels are differenced. Most calibration errors are common-mode and subtract out. Imperfect warm load emissivity, warm load temperature uncertainty, and T_{cosmic} uncertainty will all cancel out. Independent errors are the most critical and must be minimized.

Radiometer sensitivity is defined as the NEDT (noise equivalent differential temperature). It is the smallest detectable change in input brightness temperature. The wind direction nature is on the order of 1-5 deg. K. Radiometer sensitivity must be an order of magnitude smaller to be able to detect wind direction. Averaging of multiple samples enables us to improve sensitivity. Fig. A8 shows the relationship of the antenna and brightness temperatures.

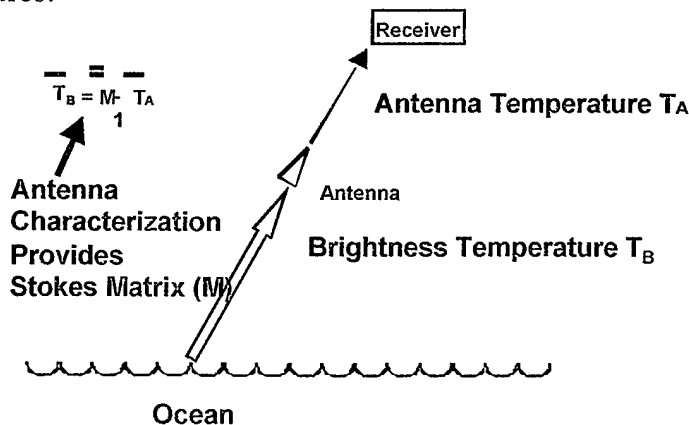


Fig. A8 Relationship of antenna and brightness temperature

The Windsat orbit was selected to be a 830 km circular orbit with a sun synchronous inclination (101.5 minute period) (similar orbit to NPOESS/DMSP). A 6:00 PM ascending node was selected to optimize performance and minimize eclipses. The orbit is usually full

sun, with an eclipse season occurring once a year for ~ 78 days. During the eclipse season, the eclipses occur every revolution and last for 17.7 minutes (17.4% of the orbit). Fig. A9 shows the orbit.

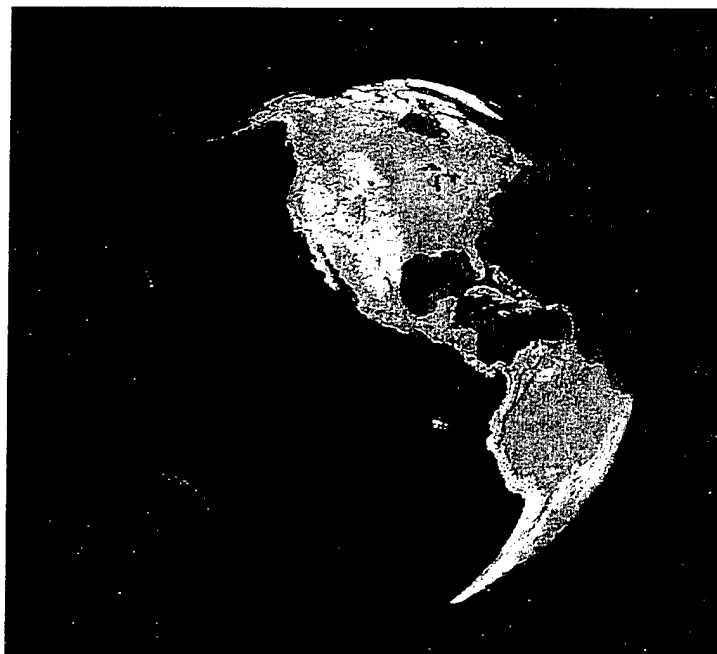


Fig. A9 Windsat sun-synchronous orbit

Appendix B.

Reflector Modeling Codes

Several computer codes are available to model reflectors. The Ohio State Reflector Code (OSUREF), developed at the Ohio State University, can be used to compute the near and far-field patterns of reflector antennas with parabolic surfaces. The code uses a combination of the GTD method and the Aperture Integration (AI) method to compute the fields. Typically, AI, also known as the Aperture Field Method, is used to compute the main beam and near sidelobes, and GTD is used to compute the wide-angle sidelobes and backlobes. Because these approaches are used, the minimum-size reflector that can be modeled is from 3 to 5 wavelengths in diameter.

The Grasp8 software from Tica in Denmark is similar to the OSUREF code but more flexible in design and implementation. It computes fields using either the PO (physical optics) method or GTD for wide angles.

Both codes were initially used to model the Windsat system and gave very similar results. Eventually, it was found that the flexibility of the Grasp8 software was far superior to the OSUREF code, and so the Grasp8 code was used to do all subsequent modeling for the system.

The angle where the PO solution degrades is dependent on the number of integration grid points used to model the reflector. With enough integration points, the PO solution can model the back lobes of the reflector and give very similar results as the GTD solution. This is important since only the PO method can be used when modeling multiple reflector interactions or interactions of the reflector with the feed or struts. The Grasp8 software is set up to handle these types of interactions. For our case of modeling the feed bench blockage on antenna sidelobes, the command line input is used to first apply currents to the reflector surface from a feed. Then the reflector surface currents are used to 'feed' the feed bench plate. The feed bench plate currents are then added to the main reflector currents to obtain the final fields that include the plate blockage.

This is a relatively easy technique for handling the complexity of multiple interactions. It was found to have some problems handling the coldsky reflector blockage of the main reflector. This was thought to be due to the closeness of the feed to the cold sky reflector and a possible breakdown of the near-field approximation that is used to determine the currents on the reflector from the feed. Tica has been informed of this problem.

OSUREF Reflector Modeling Code:

The Ohio State Reflector Code¹ was developed at the Ohio State University. It can be used to compute the near and far-field patterns of reflector antennas with parabolic surfaces. It can be used to predict the patterns of existing reflector antennas, design new reflector antennas, do radiation hazard calculations, and do EMC or coupling calculations with a reflector and small antennas.

This code uses a combination of the GTD method and the Aperture Integration (AI) method to compute the fields. Typically, AI, also known as the Aperture Field Method, is used to compute the main beam and near sidelobes, and GTD is used to compute the wide-angle sidelobes and backlobes. Because these approaches are used, the minimum-size reflector that can be modeled is from 3 to 5 wavelengths in diameter.

A general piecewise linear reflector rim shape may be used. The required input data for the feed pattern is minimized by piecewise linear pattern fitting. The feed may be

linearly polarized with any orientation or circularly polarized. A feed pattern option is available for a dominant mode horn feed in which the horn dimensions are input. Feed blockage is simulated by a physical optics model of a rectangular or a circular disk. Scattering from feed struts with circular cross section and piecewise linear axes can also be modeled. The struts can be attached to the edge of the dish or at a fixed distance inside the dish. The code also has the capability to supply input to the NEC BSC code to model more complicated scatterers such as cylinders.

Some of the limitations of the code are that the reflector surface must be parabolic. Also, the feed must be located near the focus, and the grid size used for aperture integration must be chosen sufficiently small to give a good representation of the aperture field distribution. The strut diameters should not be more than 10 wavelengths. The source of strut scattering is the geometrical optics fields from the reflector surface, other strut scattering mechanisms, such as direct feed scattering from the strut, are not modeled. This code provides significant improvement in the accuracy of near field calculations compared to the approximate EMC models previously used for reflector antennas. However, the code cannot be used to achieve accuracies greater than a dB, especially at low levels below the maximum fields.

The Reflector Code is available for a small distribution fee from:

The Ohio State University
ElectroScience Laboratory
1320 Kinnear Road
Columbus, Ohio 43212

Discussion of GRASP8 Reflector Modeling Code:

The GRASP8 Reflector Code¹ was developed at Ticra in Denmark. It can be used to compute the near and far-field patterns of reflector antennas with parabolic surfaces. It can be used to predict the patterns of existing reflector antennas, design new reflector antennas, do radiation hazard calculations, and do EMC or coupling calculations with a reflector and small antennas.

Information regarding the code is available from Ticra Engineering Consultants, www.ticra.com.

Appendix C. Comparison of OSUREF and GRASP8 Software

The GRASP8 and OSUREF codes were compared modeling the Windsat secondary patterns and gave very similar results for both the co- and the cross-pol. Figs. C1-C3 show the results.

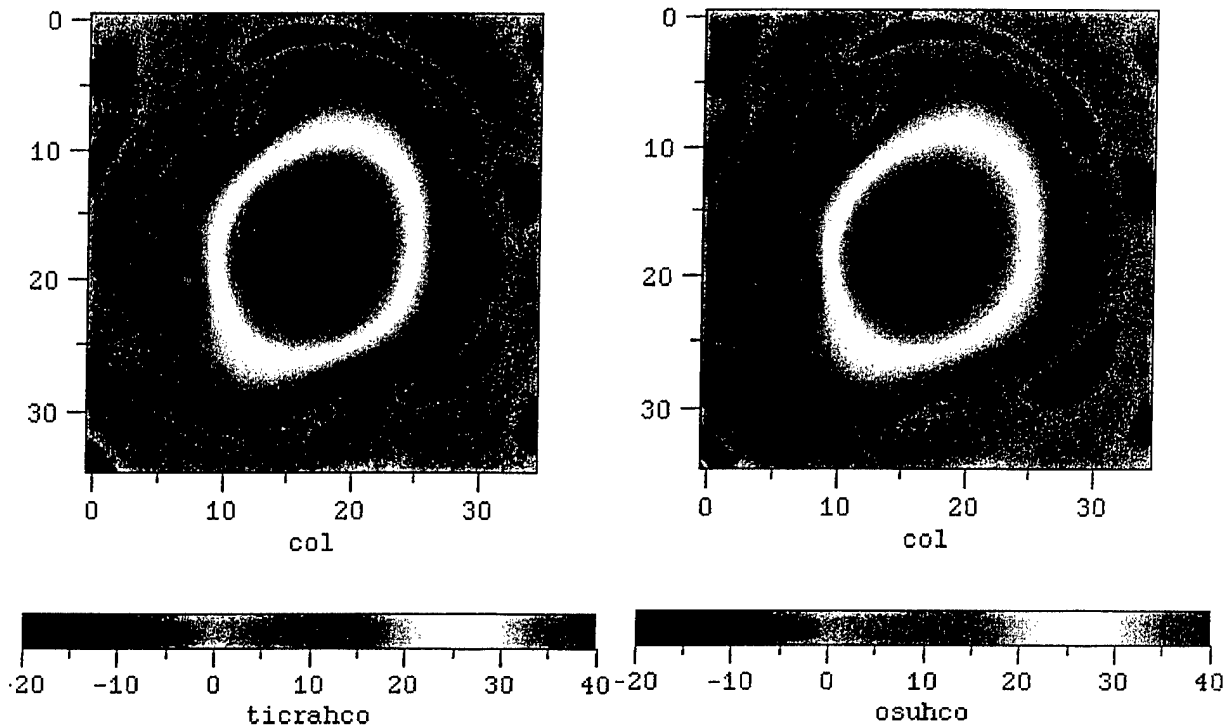


Fig. C1 Comparison of Ticra and OSUREF secondary patterns for Windsat horizontal co-pol

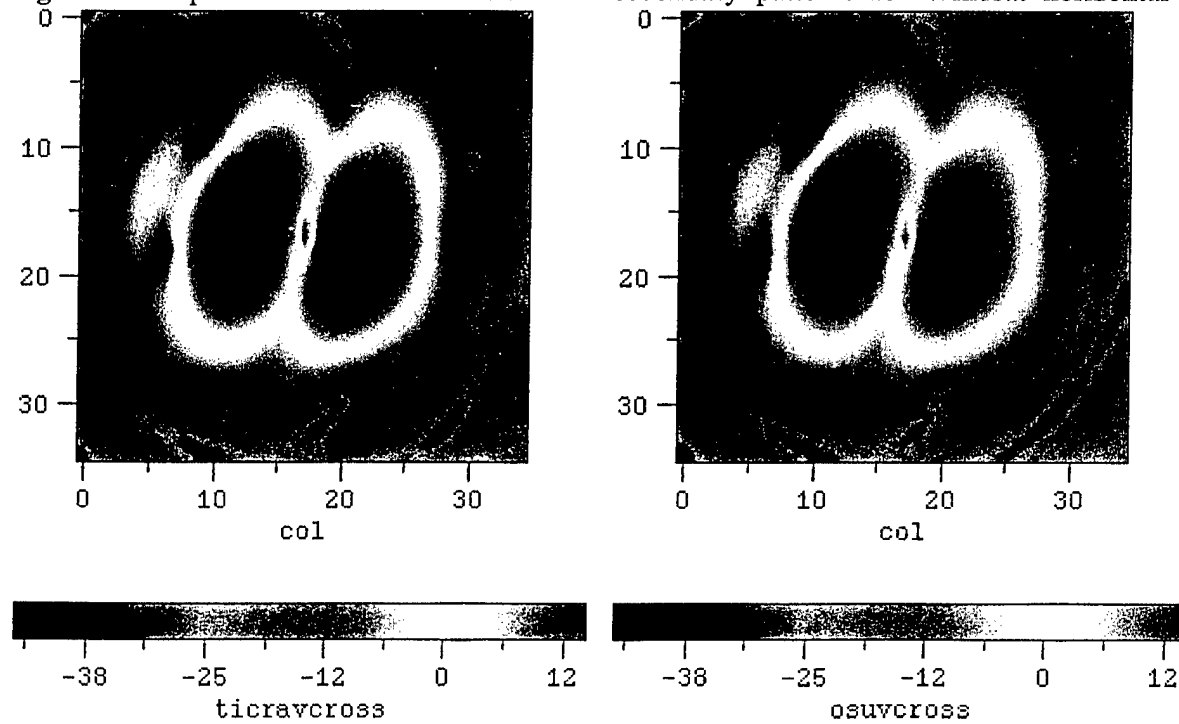
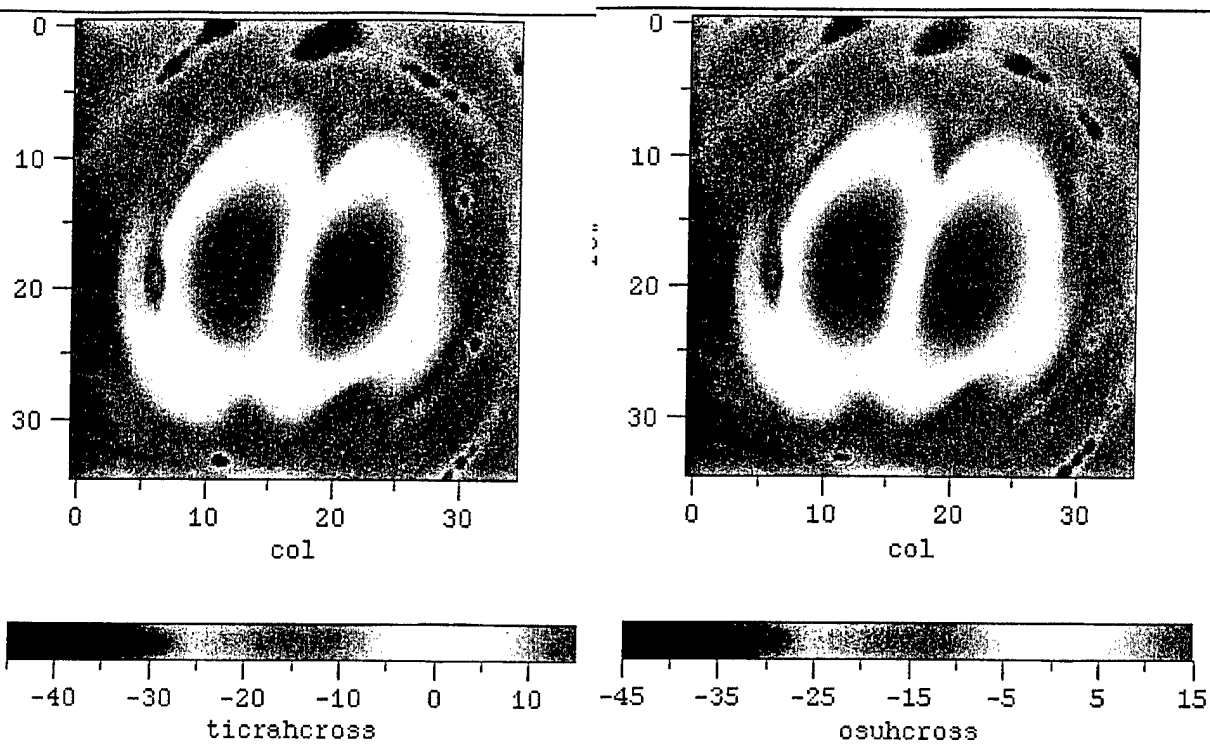


Fig. C2 Comparison of Ticra and OSUREF secondary patterns for Windsat vertical cross-pol



Offset Case

Fig. C3 Comparison of Ticra and OSUREF secondary patterns for Windsat horizontal cross-pol

Appendix D. **POE Code for +/- 45 degree Polarization**

```

C  PROGRAM NEWNRL99_WINDSAT_PM45_ANT
C  also known as raster45new.for
C  NEW RASTERPOST PROGRAM
C  UPDATED 5/11/99 TO INCLUDE PROPER ANT RANGE POLARIZATIONS
C  AND OPTION TO PROJECT BEAMS ONTO ANTENNA RANGE IN LIEU OF
C  EARTH POLARIZATION BASIS
C  modified 12/00 to change from el over az to az over el coordinate
C  transformations    WLL

C  THE PROGRAM ASSUMES INPUT CONSISTS OF:
C      CALIBRATED +/- 45 DEGREE POLARIZATAION
C      COMPLEX ANTENNA GAINS (AMPLITUDE/PHASE)
C
C  NOTE: FOR V,H ANTENNA RANGE MEASUREMENTS OR
C  FOR RIGHT/LEFT CIRCULATR POLARIZATIONS, THE USER SHOULD
C  OTHER PROGRAMS:
C      NRL99_WINDSAT_VH_ANT OR
C      NRL99_WINDSAT_RLCP_ANT

C  THE PROGRAM DOES THE FOLLOWING:
C  1. READS CALIBRATED WINDSAT +/-45 ANTENNA PATTERN MEASUREMENTS
C  2. COMPUTES THE CENTROIDS OF THE V/H BEAMS
C  3. NORMALIZES THE V/H BEAM ENERGIES
C  4. CREATES SURFACE CONTOUR PLOTS AND ANTENNA PATTERN PLOTS
C  5. PROJECTS COMPLEX ANTENNA GAIN FUNCTIONS INTO TWO GEOMETRIES:
C      1. EARTH SURFACE
C      2. ANTENNA RANGE
C  6. COMPUTES STOKES PARAMETER COUPLING MATRIX FOR THE SELECTED
C      GEOMETRY
C
C  IMPLICIT REAL*8(A,B,D-H,O-Z)
C  IMPLICIT COMPLEX*16(C)

C  DIMENSION AK(3),HE(3),VE(3),VT(3),HT(3),PE(3),AME(3)

C  DIMENSION AK0VPOL(3),AV0VPOL(3),AH0VPOL(3),AK0HPOL(3),
+      AV0HPOL(3),AH0HPOL(3),
+      AV0VPOLR(3),AV0HPOLR(3),AH0VPOLR(3),AH0HPOLR(3)
C  DIMENSION CEVPOL(3),CEHPOL(3),CEVPOLT(3),CEHPOLT(3)

C  DIMENSION XGVEVA(201),XGHEVA(201),XGUEVA(201),XG4EVA(201)
C  DIMENSION XGVEHA(201),XGHEHA(201),XGUEHA(201),XG4EHA(201)
C  DIMENSION XGVEUA(201),XGHEUA(201),XGUEUA(201),XG4EUA(201)
C  DIMENSION XGVE4A(201),XGHE4A(201),XGUE4A(201),XG4E4A(201)
C  DIMENSION zr(3),zm(3),zx(3),zy(3)
C  DIMENSION wBHTQ_BET(101)
C  DIMENSION wBHTQ_ALP(101)
C  DIMENSION wBVTP_BET(101)
C  DIMENSION wBVTP_ALP(101)

C  DIMENSION vcomag(1,75,75)
C  DIMENSION vcophase(1,75,75),vcrossmag(1,75,75)
C  DIMENSION vcrossphase(1,75,75),hcomag(1,75,75)
C  DIMENSION hcophase(1,75,75),hcrossmag(1,75,75)
C  DIMENSION hcrossphase(1,75,75)
C  DIMENSION aziangle(1,75,75),elevangle(1,75,75)
C  DIMENSION elr(81),azrx(81),g4v(75,75),g4h(75,75)
C  DIMENSION c2vcomag(1,75,75),c2hcrossmag(1,75,75)
C  DIMENSION c2hcomag(1,75,75),c2vcrossmag(1,75,75)

C  DIMENSION azcenterv(51),elcenterv(51)
C  DIMENSION azcenterh(51),elcenterh(51)

```

CHARACTER*1 ANS_ROT,ANS_DUM

CHARACTER*20 logname

CHARACTER *50 FILEIN,FILEOUT,FILEOUT2

CHARACTER*2 aa(100)

DATA (aa(j), j=1,50) /'1','2','3','4','5',

1 '6','7','8','9','10',

1 '11','12','13','14','15','16','17','18','19','20',

1 '21','22','23','24','25','26','27','28','29','30',

1 '31','32','33','34','35','36','37','38','39','40',

1 '41','42','43','44','45','46','47','48','49','50'/

OPEN(UNIT=55,FILE='inputfile.txt',STATUS='OLD')

OPEN(UNIT=66,FILE='outlog2raster45.txt',STATUS='UNKNOWN')

OPEN(UNIT=50,FILE='outlog3raster45.txt',STATUS='UNKNOWN')

OPEN(UNIT=36,FILE='centroid45file',STATUS='UNKNOWN')

OPEN(UNIT=31,FILE='powernormfactors.txt',STATUS='UNKNOWN')

READ(55,5567)logname

5567 FORMAT(A20)

C ifreqnum= number of frequencies

READ(55,*)ifreqnum

C numaz, numel = enter number of grid points, az & elev.

C these don't have to be equal

READ(55,*)numaz,numel

C ginc pixel size of data in degrees

C ex. .03200 deg for 37 GHz

READ(55,*)ginc

C freq=center frequency; freqinc=freq increment GHz

C (for elev. correction), ex. 37, .2

C not used in this program

READ(55,*)freq

READ(55,*)freqinc

freqprint=freq-(freqinc*(ifreqnum+1)/2)

C hrot=amount of rotation to data

C 0 for none

C note: all rotations now occur in the pre-processor

hrot=0.0

C rotang is the pre-processor rotation

READ(55,*)rotang

C*****

C Shiftd is the angle between the horiz. and vertical source antennas

C not used by raster45, rastercp, or rastervh.for

READ(55,*)shiftd

C*****

C azimuth encoder offset

READ(55,*)azencoder

C elevation encoder offset

READ(55,*)elencoder

WRITE(36,*) Centroid information '

WRITE(36,*) azimuth encoder offset input: ',SNGL(azencoder)

WRITE(36,*) elevation encoder offset input: ',SNGL(elencoder)

WRITE(36,*)note: these are from inputfile.txt '

WRITE(36,*)

WRITE(36,*) Windsat coordinate

1 Raw Encoder Values '

WRITE(36,*) system '

WRITE(36,*) freq. azimuth elevation

1 azimuth elevation '

WRITE(36,*) GHz ang. ang.

1 ang. ang. '

WRITE(36,*) deg. deg.

1 deg. deg. '

WRITE(36,*)

C azz is the azimuth half angle of data in deg.

C ell is the elevation half angle of data in deg.

azz=(numaz-1)*ginc/2.

ell=(numel-1)*ginc/2.


```

C*****
C   matout is written in this program to contain all Stokes data
C   for all frequencies, temporary file used as input to freqaverage.for
C   OPEN(UNIT=29,FILE='matout',STATUS='UNKNOWN')
C   centroidfile contains the centroid information,
C   only the averaged centroid az and el centers are used
C   the other centroid information is in the file to check if data is OK
C       OPEN(UNIT=33,FILE='centroidprefile',STATUS='OLD')
C       READ(33,*)
C       READ(33,*)
C
C       READ(33,*)azcenter,elcenter
C       CLOSE(33)
C*****
C   DEFINE CONSTANTS
C       PIE = DACOS(-1.D0)
C       RAD = PIE/180.D0
C       FPIE = PIE * 4.D0
C       ROOT2 = DSQRT(2.D0)
C       THREEDB = 10.D0 * DLOG10(2.D0)
C   THREEDB=0.0D0
C   WRITE(*,*) 'THREEDB:',SNGL(THREEDB)
C   beginning of the frequency loop
C       ifreqloop=0
339  CONTINUE
C       ifreqloop=ifreqloop+1
C       freqprint=freqprint+freqinc
C       WRITE(50,*)'*****'
C       WRITE(50,*)'*****FREQUENCY # ',ifreqloop
C       WRITE(50,*)'*****'
C       WRITE(66,*)'*****'
C       WRITE(66,*)'*****FREQUENCY # ',ifreqloop
C       WRITE(66,*)'*****'
C   INITIALIZE TO ZERO
C       DO k=1,201
C           XGVEVA(k)=0.D0
C           XGHEVA(k)=0.D0
C           XGUEVA(k)=0.D0
C           XG4EVA(k)=0.D0
C           XGVEHA(k)=0.D0
C           XGHEHA(k)=0.D0
C           XGUEHA(k)=0.D0
C           XG4EHA(k)=0.D0
C           XGVEUA(k)=0.D0
C           XGHEUA(k)=0.D0
C           XGUEUA(k)=0.D0
C           XG4EUA(k)=0.D0
C           XGVE4A(k)=0.D0
C           XGHE4A(k)=0.D0
C           XGUE4A(k)=0.D0
C           XG4E4A(k)=0.D0
C       ENDDO
C*****
C*****
C   READ IN ANTENNA RANGE MEASUREMENTS AND INTERPOLATE DATA
C   TO FIXED AZ,EL GRID
C
C       IFRQ = 1
C       IFLAG = 0
C*****
C
C       WRITE(*,*)'write out array data?'
C       READ(55,*)iprint
C       iprint=0
C       WRITE(*,*) 'ENTER RANGE DATA INPUT FILE NAME (+.EXT):'
C       READ(55, '(A)') FILEIN
C       OPEN(UNIT=11,FILE='.\output\rangeout\\aa(ifreqloop),

```

```

1 STATUS='OLD')
C*****
C set up the pointing of the beam based on the center of
C the data taken (need to add in offset from centroiding in
C the Rasterpre program
C*****

      nnf=ifreqloop
      WRITE(66,*)'azcenter, elcenter used'
      WRITE(66,*)azcenter, elcenter
      amin=azcenter-azz
      amax=azcenter+azz
      elmin=elcenter-ell
      elmax=elcenter+ell
      ainc=azz/((numaz-1)/2)
      WRITE(66,*)'data increment is: ',ainc
C      Positive elevation now
C refers to up (opposite to right hand rule,)
      ang=amax
      angel=elmax
      DO jj=1,numaz
        azrx(jj)=ang
        ang=ang-ainc
      ENDDO
      DO jj=1,numel
        elr(jj)=angel
        angel=angel-ainc
      ENDDO
      ifreq=1

C*****
C READ IN RANGE DATA
C note: angles read in through these statements are ignored
C*****
C*****
      numfreq=1
      WRITE(66,*)' gone into reading range data input'
      WRITE(66,*)'OSUREF interpolated data'
      READ(11,*)
      DO j=1,numaz
        DO k=1,numel
          DO i=1,numfreq
C          READ(11,*)azangle(i,j,k),elevangle(i,j,k),
C 1 vcomag(i,j,k),
C 1 vcophase(i,j,k),vcrossmag(i,j,k),vcrossphase(i,j,k),
C 1 hcomag(i,j,k),
C 1 hcophase(i,j,k),hcrossmag(i,j,k),hcrossphase(i,j,k)
          READ(11,*,END=999)azangle(i,j,k),elevangle(i,j,k),
1 vcomag(i,j,k),vcrossmag(i,j,k),hcomag(i,j,k),
1 hcrossmag(i,j,k)

          VCOMAG(I,J,K) = VCOMAG(I,J,K) - THREEDB
          HCOMAG(I,J,K) = HCOMAG(I,J,K) - THREEDB
          VCROSSMAG(I,J,K) = VCROSSMAG(I,J,K) - THREEDB
          HCROSSMAG(I,J,K) = HCROSSMAG(I,J,K) - THREEDB

          READ(11,*,END=999)azangle(i,j,k),elevangle(i,j,k),
1 vcophase(i,j,k),vcrossphase(i,j,k),hcophase(i,j,k),
1 hcrossphase(i,j,k)

          IF(hcrossphase(i,j,k).GE.180.)THEN
            hcrossphase(i,j,k)=hcrossphase(i,j,k)-360.
          ENDIF
        ENDDO
      ENDDO
      ENDDO
999 CONTINUE
      IFREQ = 1

```

```

112      FORMAT(I13,F10.3,F10.3,4(F11.4,F11.3))
      CLOSE(11)
      WRITE(66,*) ' COMPLETED READING RANGE INPUT DATA:'

C*****
C Need to convert the range az/el values to the spacecraft
C az/el values given by Bill Purdy coordinate transformations
C This amounts to a small angular difference but must be taken
C account of in the Stokes matrix, EIA and SCAN angles.
C The PRA is taken account of by the inputfile.txt input angle
C*****
      DO j=1,numaz
      DO k=1,numel
      DO i=1,numfreq
        azianglez= aziangle(i,j,k)
        elevanglez= elevangle(i,j,k)
      CALL PURDYTRANSFORM(azianglez,elevanglez,azianglenew,
1 elevanglenew,freq)

        aziangle(i,j,k)= azianglenew
        elevangle(i,j,k)= elevanglenew

      ENDDO
      ENDDO
      ENDDO

C*****
C Reverse sign of elevation and azimuth angles
C to match up with Gene Poe coordinate system
C*****
      DO j=1,numaz
      DO k=1,numel
      DO i=1,numfreq

        aziangle(i,j,k)= -aziangle(i,j,k)
        elevangle(i,j,k)= -elevangle(i,j,k)

      ENDDO
      ENDDO
      ENDDO

C*****
C First convert to complex numbers to form complex conjugate
C*****
      DPR=57.29578
      PI=3.14159265359
      DPI=6.28318530718

      DO j=1,numaz
      DO k=1,numel
      DO i=1,numfreq

C convert to complex numbers here:
      VERT=10**(vcomag(i,j,k)/20.)
      PHI1=(vcophase(i,j,k)/360.)*DPI
      IF (PHI1.LT.0) PHI1=PHI1+DPI
      vxx=VERT/((1+(DTAN(PHI1))**2)**.5)
      vyy=vxx*DTAN(PHI1)
      ANGR=PHI1*DPR
      IF(ANGR.GT.90. .AND. ANGR.LE.180.)vxx=-vxx
      IF(ANGR.GT.90. .AND. ANGR.LE.180.)vyy=-vyy
      IF(ANGR.GT.180. .AND. ANGR.LE.270.)vxx=-vxx
      IF(ANGR.GT.180. .AND. ANGR.LE.270.)vyy=-vyy
      c2vcomag(i,j,k)=CMPLX(vxx,vyy)

      VERT=10**(vcrossmag(i,j,k)/20.)
      PHI1=(vcrossphase(i,j,k)/360.)*DPI
      IF (PHI1.LT.0) PHI1=PHI1+DPI

```

```

vxx=VERT/((1+(DTAN(PHI1))**2)**.5)
vyy=vxx*DTAN(PHI1)
ANGR=PHI1*DPR
IF(ANGR.GT.90. .AND. ANGR.LE.180.)vxx=-vxx
IF(ANGR.GT.90. .AND. ANGR.LE.180.)vyy=-vyy
IF(ANGR.GT.180. .AND. ANGR.LE.270.)vxx=-vxx
IF(ANGR.GT.180. .AND. ANGR.LE.270.)vyy=-vyy
c2vcrossmag(i,j,k)=CMPLX(vxx,vyy)

```

```

VERT=10**(hcomag(i,j,k)/20.)
PHI1=(hcophase(i,j,k)/360.)*DPI
IF (PHI1.LT.0) PHI1=PHI1+DPI
vxx=VERT/((1+(DTAN(PHI1))**2)**.5)
vyy=vxx*DTAN(PHI1)
ANGR=PHI1*DPR
IF(ANGR.GT.90. .AND. ANGR.LE.180.)vxx=-vxx
IF(ANGR.GT.90. .AND. ANGR.LE.180.)vyy=-vyy
IF(ANGR.GT.180. .AND. ANGR.LE.270.)vxx=-vxx
IF(ANGR.GT.180. .AND. ANGR.LE.270.)vyy=-vyy
c2hcomag(i,j,k)=CMPLX(vxx,vyy)

```

```

VERT=10**(hcrossmag(i,j,k)/20.)
PHI1=(hcrossphase(i,j,k)/360.)*DPI
IF (PHI1.LT.0) PHI1=PHI1+DPI
vxx=VERT/((1+(DTAN(PHI1))**2)**.5)
vyy=vxx*DTAN(PHI1)
ANGR=PHI1*DPR
IF(ANGR.GT.90. .AND. ANGR.LE.180.)vxx=-vxx
IF(ANGR.GT.90. .AND. ANGR.LE.180.)vyy=-vyy
IF(ANGR.GT.180. .AND. ANGR.LE.270.)vxx=-vxx
IF(ANGR.GT.180. .AND. ANGR.LE.270.)vyy=-vyy
c2hcrossmag(i,j,k)=CMPLX(vxx,vyy)

```

ENDDO

ENDDO

ENDDO

WRITE(66,*)'finished conversion to complex numbers'

C GOTO 5566

C*****

C need to convert data from complex conjugate before rotation

C*****

DO j=1,numaz

DO k=1,numel

DO i=1,numfreq

c2vcomag(i,j,k)=DCONJG(c2vcomag(i,j,k))

c2vcrossmag(i,j,k)=DCONJG(c2vcrossmag(i,j,k))

c2hcomag(i,j,k)=DCONJG(c2hcomag(i,j,k))

c2hcrossmag(i,j,k)=DCONJG(c2hcrossmag(i,j,k))

ENDDO

ENDDO

ENDDO

C*****

C do rotation

C remember that vco and hcross are rotated together,

C and hco and vcross are rotated together

C*****

C WRITE(66,*)'enter amount of rotation to data before '

C WRITE(66,*)'main processing, in deg., 0 for none'

C READ(55,*)hrot

SINPT=DSIND(hrot)

COSPPT=DCOSD(hrot)

DO j=1,numaz

DO k=1,numel

DO i=1,numfreq

ca1=c2vcomag(i,j,k)

ca2=c2hcrossmag(i,j,k)

ca3=c2hcomag(i,j,k)

ca4=c2vcrossmag(i,j,k)

```

c2vcomag(i,j,k)=COSPT*ca1+SINPT*ca2
c2hcrossmag(i,j,k)=-SINPT*ca1+COSPT*ca2

c2vcrossmag(i,j,k)=COSPT*ca4+SINPT*ca3
c2hcomag(i,j,k)=-SINPT*ca4+COSPT*ca3

ENDDO
ENDDO
ENDDO
C*****
C need to convert data back to complex conjugate after rotation
C*****
DO j=1,numaz
DO k=1,numel
DO i=1,numfreq
c2vcomag(i,j,k)=DCONJG(c2vcomag(i,j,k))
c2vcrossmag(i,j,k)=DCONJG(c2vcrossmag(i,j,k))
c2hcomag(i,j,k)=DCONJG(c2hcomag(i,j,k))
c2hcrossmag(i,j,k)=DCONJG(c2hcrossmag(i,j,k))
ENDDO
ENDDO
ENDDO
C*****
C convert data back to magnitude and phase, again
C*****
5566 CONTINUE
DO j=1,numaz
DO k=1,numel
DO i=1,numfreq

c1=c2vcomag(i,j,k)
zmagb1=20*LOG10(SQRT(DIMAG(c1)**2+DREAL(c1)**2))
phaseb1=DATAN2D(DIMAG(c1),DREAL(c1))
IF(phaseb1.GT.180.) phaseb1=phaseb1-360.
vcomag(i,j,k)=zmagb1
vcophase(i,j,k)=phaseb1

c2=c2vcrossmag(i,j,k)
zmagb2=20*LOG10(SQRT(DIMAG(c2)**2+DREAL(c2)**2))
phaseb2=DATAN2D(DIMAG(c2),DREAL(c2))
IF(phaseb2.GT.180.) phaseb2=phaseb2-360.
vcrossmag(i,j,k)=zmagb2
vcrossphase(i,j,k)=phaseb2

c3=c2hcomag(i,j,k)
zmagb3=20*LOG10(SQRT(DIMAG(c3)**2+DREAL(c3)**2))
phaseb3=DATAN2D(DIMAG(c3),DREAL(c3))
IF(phaseb3.GT.180.) phaseb3=phaseb3-360.
hcomag(i,j,k)=zmagb3
hcophase(i,j,k)=phaseb3

c4=c2hcrossmag(i,j,k)
zmagb4=20*LOG10(SQRT(DIMAG(c4)**2+DREAL(c4)**2))
phaseb4=DATAN2D(DIMAG(c4),DREAL(c4))
IF(phaseb4.GT.180.) phaseb4=phaseb4-360.
hcrossmag(i,j,k)=zmagb4
hcrossphase(i,j,k)=phaseb4

ENDDO
ENDDO
ENDDO
5993 CONTINUE
C*****
C*****
C*****
C*****
C*****
C change from dB to amplitude:

```

```

C      DO k=1,numel
      DO j=1,numaz
      DO i=1,numfreq
vcomag(i,j,k)=10**(vcomag(i,j,k)/20.)
vcrossmag(i,j,k)=10**(vcrossmag(i,j,k)/20.)
hcomag(i,j,k)=10**(hcomag(i,j,k)/20.)
hcrossmag(i,j,k)=10**(hcrossmag(i,j,k)/20.)

      ENDDO
      ENDDO
      ENDDO

155  CONTINUE

C*****
C  COMPUTE ANTENNA BORESIGHT (by centroiding the beam)
C*****
C
      GVTPKX = 0.D0
      GVTPKY = 0.D0
      GHTQKX = 0.D0
      GHTQKY = 0.D0

      GVTPK0 = 0.D0
      GHTQK0 = 0.D0

      DO J=1,numaz
C      BET  = AZRX(J)
C      ACBET = DCOSD(BET)
C      ASBET = DSIND(BET)
          numeladd=(numel-1)/2
C      numeladd=1
          numelcc=((numel-1)/2)+1
      DO I=numelcc-numeladd,numelcc+numeladd
C      DO I=1,numel
          BET  = aziangle(1,j,i)
          ACBET = DCOSD(BET)
          ASBET = DSIND(BET)
C      ALP  = ELR(I)
          ALP  = elevangle(1,j,i)
          ACALP = DCOSD(ALP)
          ASALP = DSIND(ALP)

C
C      DEFINE ANTENNA BORESIGHT AK, el over az
C      AK(1) = -ACALP * ASBET
C      AK(2) =  ASALP
C      AK(3) =  ACALP * ACBET
C      DEFINE ANTENNA RANGE TRANSMIT VERTICAL POL VT, el over az
C      VT(1) =  ASALP * ASBET
C      VT(2) =  ACALP
C      VT(3) = -ASALP * ACBET
C      DEFINE ANTENNA RANGE TRANSMIT HORIZONTAL POL HT, el over az
C      HT(1) =  ACBET
C      HT(2) =  0.D0
C      HT(3) =  ASBET
C      these are the az over el matrix values, WLL 11/00 corrected
C      AK(1) = - ASBET
C      AK(2) =  ASALP * ACBET
C      AK(3) =  ACALP * ACBET
C      VT(1) =  0.D0
C      VT(2) =  ACALP
C      VT(3) = -ASALP
C      HT(1) =  ACBET
C      HT(2) =  ASALP * ASBET
C      HT(3) =  ACALP * ASBET

C*****
C  GET COMPLEX ANTENNA GAIN FUNCTIONS
      GVTPA_A=vcomag(1freq,j,i)

```

```
GHTPA_A=vcrossmag(ifreq,j,i)
GVTQA_A=hcrossmag(ifreq,j,i)
GHTQA_A=hcomag(ifreq,j,i)
```

```
PVTPA_P=vcophase(ifreq,j,i)
PHTPA_P=vcrossphase(ifreq,j,i)
PVTQA_P=hcrossphase(ifreq,j,i)
PHTQA_P=hcophase(ifreq,j,i)
```

```
CPVTPA = DCMLX(DCOSD(PVTPA_P),DSIND(PVTPA_P))
CPHTPA = DCMLX(DCOSD(PHTPA_P),DSIND(PHTPA_P))
CPVTQA = DCMLX(DCOSD(PVTQA_P),DSIND(PVTQA_P))
CPHTQA = DCMLX(DCOSD(PHTQA_P),DSIND(PHTQA_P))
```

C FORM COMPLEX EFFECTIVE HEIGHT VECTORS

```
DO L=1,3
CEVPOL(L) = GVTQA_A*CPVTQA * VT(L) + GHTQA_A*CPHTQA * HT(L)
CEHPOL(L) = GVTQA_A*CPVTQA * VT(L) + GHTQA_A*CPHTQA * HT(L)
ENDDO
```

C DEFINE WEIGHTING FOR +/- 45 POLS

```
WTVA = 0.D0
WTHA = 0.D0
DO L=1,3
WTVA = WTVA + CDABS(CEVPOL(L))**2
WTHA = WTHA + CDABS(CEHPOL(L))**2
ENDDO
```

C this scaling if for elevation over azimuth (wrt cos(elevation))

```
C GVTPKX = GVTPKX + AK(1)*WTVA *ACALP
C GVTPKY = GVTPKY + AK(2)*WTVA *ACALP
C GHTQKX = GHTQKX + AK(1)*WTHA *ACALP
C GHTQKY = GHTQKY + AK(2)*WTHA *ACALP
C GVTPK0 = GVTPK0 + WTVA *ACALP
C GHTQK0 = GHTQK0 + WTHA *ACALP
```

C this scaling if for azimuth over elevation (wrt cos(azimuth))

```
C modified 11/00 WLL
GVTPKX = GVTPKX + AK(1)*WTVA *ACBET
GVTPKY = GVTPKY + AK(2)*WTVA *ACBET
GHTQKX = GHTQKX + AK(1)*WTHA *ACBET
GHTQKY = GHTQKY + AK(2)*WTHA *ACBET
GVTPK0 = GVTPK0 + WTVA *ACBET
GHTQK0 = GHTQK0 + WTHA *ACBET
```

```
ENDDO
ENDDO
```

```
BVTPKX = GVTPKX/GVTPK0
BVTPKY = GVTPKY/GVTPK0
BVTPKZ = DSQRT(1.D0 - BVTPKX**2 - BVTPKY**2)
```

```
BHTQKX = GHTQKX/GHTQK0
BHTQKY = GHTQKY/GHTQK0
BHTQKZ = DSQRT(1.D0 - BHTQKX**2 - BHTQKY**2)
```

C CALL RAST TO FIND EL,AZ ASSOCIATED WITH KX,KY FOR V POL

```
AK(1) = BVTPKX
AK(2) = BVTPKY
AK(3) = BVTPKZ
```

```
CALL RAST(AK,BVTP_ALP,BVTP_BET,VT,HT)
```

C DEFINE BORESIGHT AND +45 AND -45 -POL UNIT VECTORS

```
DO L=1,3
AK0VPOL(L) = AK(L)
AV0VPOL(L) = ( VT(L) + HT(L) )/ROOT2
AH0VPOL(L) = ( VT(L) - HT(L) )/ROOT2
ENDDO
```

```
WRITE(50,3342)freqprint,
```

```

1  SNGL(-BVTP_BET),SNGL(-BVTP_ALP),
1  SNGL(-BVTP_BET+azencoder),SNGL(-BVTP_ALP+elencoder)
  WRITE(36,3342)freqprint,
1  SNGL(-BVTP_BET),SNGL(-BVTP_ALP),
1  SNGL(-BVTP_BET+azencoder),SNGL(-BVTP_ALP+elencoder)
3342  FORMAT(7H+45-POL,2x,F6.3,4(F13.4))

C  CALL RAST TO FIND EL,AZ ASSOCIATED WITH KX,KY FOR H POL
  AK(1) = BHTQKX
  AK(2) = BHTQKY
  AK(3) = BHTQKZ
C*****
  THETA0 = 45.D0
  ZR(1) = 0.0
  ZR(2) = DSIND(THETA0)
  ZR(3) = -DCOSD(THETA0)

  EIASAT=DACOSD(ZR(1)*AK(1)+ZR(2)*AK(2)+ZR(3)*AK(3))
C  earth radius in km
  r1=6371
C  satellite height in km
  r2x=830.
  r2=r1+r2x
  eia=(DASIND(SIND(180-eiasat)/r1*r2))
  WRITE(50,*)'ak', SNGL(AK(1)),SNGL(AK(2)),SNGL(AK(3))
  WRITE(50,*)'eia 'eia 'eia
C  WRITE(6,*)'eia 'eia
C***** now find scan angle *****
C  first set x vector
  Zx(1) = 1.0D0
  Zx(2) = 0.0D0
  Zx(3) = 0.0D0
C  do crossproduct of z x x = y to find new y vector
  XP2=Zr(2)*zx(3)
  XP1=Zr(3)*zx(2)

  YP2=Zr(3)*zx(1)
  YP1=Zr(1)*zx(3)

  ZP2=Zr(1)*zx(2)
  ZP1=Zr(2)*zx(1)

  zy(1)=(XP2-XP1)
  zy(2)=(YP2-YP1)
  zy(3)=(ZP2-ZP1)
C  WRITE(6,*)'zx',SNGL(zx(1)),SNGL(zx(2)),SNGL(zx(3))
C  WRITE(6,*)'zy',SNGL(zy(1)),SNGL(zy(2)),SNGL(zy(3))
C  WRITE(6,*)'zr',SNGL(zr(1)),SNGL(zr(2)),SNGL(zr(3))
C  WRITE(6,*)'ak',SNGL(ak(1)),SNGL(ak(2)),SNGL(ak(3))
  zm(1)=ak(1)*zx(1)+ak(2)*zx(2)+ak(3)*zx(3)
  zm(2)=ak(1)*zy(1)+ak(2)*zy(2)+ak(3)*zy(3)
  zm(3)=ak(1)*zr(1)+ak(2)*zr(2)+ak(3)*zr(3)
C  WRITE(6,*)
C  WRITE(6,*)'zm ',SNGL(zm(1)),SNGL(zm(2)),SNGL(zm(3))
  THETAYVC=DACOSD((zm(3))/SQRT((zm(1))**2+
1 (zm(2))**2+(zm(3))**2))
  PHIYVC=DATAN2D((zm(2)),(zm(1)))
C  WRITE(6,*)'these should be the same, nadir angle'
C  WRITE(6,*)'eiasat,THETAYVC
C  write(6,*)'scan angle ',PHIYVC
  write(50,*)'scan angle ',PHIYVC
C*****

  CALL RAST(AK,BHTQ_ALP,BHTQ_BET,VT,HT)

C  DEFINE BORESIGHT AND +45 AND -45 -POL UNIT VECTORS
  DO L=1,3
  AK0HPOL(L) = AK(L)

```



```

      AV0HPOL(L) = ( VT(L) + HT(L) )/ROOT2
      AH0HPOL(1) = ( VT(L) - HT(L) )/ROOT2
      ENDDO
C   save centroid information so can print out average later
      wBHTQ_BET(ifreqloop)=BHTQ_BET
      wBHTQ_ALP(ifreqloop)=BHTQ_ALP
      wBVTP_BET(ifreqloop)=BVTP_BET
      wBVTP_ALP(ifreqloop)=BVTP_ALP

      WRITE(50,3341)freqprint,SNGL(-BHTQ_BET),SNGL(-BHTQ_ALP),
1  SNGL(-BHTQ_BET+azencoder),SNGL(-BHTQ_ALP+elencoder)

      WRITE(36,3341)freqprint,SNGL(-BHTQ_BET),SNGL(-BHTQ_ALP),
1  SNGL(-BHTQ_BET+azencoder),SNGL(-BHTQ_ALP+elencoder)
3341  FORMAT(7H-45-POL,2x,F6.3,4(F13.4))

C*****
C   COMPLETED CENTROID COMPUTATIONS
C*****

C   WRITE(*,*) ' TO PROJECT ANTENNA BEAMS ONTO EARTH, ENTER 1:'
C   WRITE(*,*) '           ANTENNA BEAMS ONTO ANT RANGE,  2:'
C   WRITE(*,*) ' ENTER SELECTION:'
C   READ(55,*) IANS_PRO
      IANS_PRO=1
      POLROT = 0.D0
      ACPOLROT = DCOSD(POLROT)
      ASPOLROT = DSIND(POLROT)

      IF(IANS_PRO.EQ.1)THEN
C   COMPUTE POLARIZATION ROTATION OF BEAM WRT EARTH POLARIZATION
      WRITE(50,*) ' WE PROJECT ANTENNA BEAMS ONTO EARTH SURFACE:'

      CALL GEODETIC(AK0VPOL,VE,HE)
      DO L=1,3
      PE(L)=( VE(L) + HE(L) )/ROOT2
      AME(L)=( VE(L) - HE(L) )/ROOT2
      ENDDO

      ACPOLROTV = 0.D0
      ACPOLROTH = 0.D0
      DO L=1,3
      ACPOLROTV = ACPOLROTV + AV0VPOL(L)*PE(L)
      ACPOLROTH = ACPOLROTH + AV0VPOL(L)*AME(L)
      ENDDO

      POLROTV = DATAN2D(ACPOLROTH,ACPOLROTV)
      WRITE(50,*) ' POL ROT FOR + 45 POL:',SNGL(POLROTV)

      CALL GEODETIC(AK0HPOL,VE,HE)

      ACPOLROTV = 0.D0
      ACPOLROTH = 0.D0
      DO L=1,3
      ACPOLROTV = ACPOLROTV + AV0HPOL(L)*PE(L)
      ACPOLROTH = ACPOLROTH + AV0HPOL(L)*AME(L)
      ENDDO

      POLROTH = DATAN2D(ACPOLROTH,ACPOLROTV)
      WRITE(50,*) ' POL ROT FOR -45 POL:',SNGL(POLROTH)

C   WRITE(*,*) ' WANT TO ROTATE FEEDHORN TO ALIGN THE EARTH AND '
C   WRITE(*,*) ' ANTENNA POLARIZATION BASIS? (Y/N):'
C   READ(55, '(A)') ANS_ROT
      ANS_ROT='Y'
      IF(ANS_ROT.EQ.'Y'.OR.ANS_ROT.EQ.'y')THEN
C   WRITE(*,*) ' WANT TO USE AVERAGE OF ABOVE ROTATION ANGLE?(Y/N):'
C   READ(55, '(A)') ANS_DUM
      ANS_DUM='n'

```

```

      IF(ANS_DUM.EQ.'Y'.OR.ANS_DUM.EQ.'y')THEN
        POLROT = 0.5D0*(POLROT $\bar{V}$  + POLROTH)
      ELSE
C      WRITE(*,*) ' Input Pol Offset in deg:'
        POLROT=0.D0
      ENDIF

      WRITE(50,*) ' WE ROTATE FEEDHORN (D):',SNGL(POLROT)
      ACPOLROT = DCOSD(POLROT)
      ASPOLROT = DSIND(POLROT)

      ELSE
      WRITE(50,*) ' WE DO NOT ROTATE FEEDHORN TO ALIGN EARTH AND '
      WRITE(50,*) ' ANTENNA RANGE POLARIZATION BASIS:'

      ENDIF
    ENDIF

C
C*****
C      NEXT WE FIND THE NORMALIZATION FACTORS FOR THE ANTENNA
C      BEAM ENERGIES

      ANORM_V = 0.D0
      ANORM_H = 0.D0
      CORTH0 = DCMPLX(0.D0,0.D0)

      DO j=1,numaz
C      BET = AZRX(J)
C      ACBET = DCOSD(BET)
C      ASBET = DSIND(BET)

      DO I=numelcc-numeladd,numelcc+numeladd
C      DO I=1,numel
        BET = aziangle(1,j,i)
        ACBET = DCOSD(BET)
        ASBET = DSIND(BET)
C      ALP = ELR(I)
        ALP = elevangle(1,j,i)
        ACALP = DCOSD(ALP)
        ASALP = DSIND(ALP)

C      DEFINE ANTENNA BORESIGHT AK
C      AK(1) = -ACALP * ASBET
C      AK(2) = ASALP
C      AK(3) = ACALP * ACBET
C      DEFINE ANTENNA RANGE TRANSMIT VERTICAL POL VT
C      VT(1) = ASALP * ASBET
C      VT(2) = ACALP
C      VT(3) = -ASALP * ACBET
C      DEFINE ANTENNA RANGE TRANSMIT HORIZONTAL POL HT
C      HT(1) = ACBET
C      HT(2) = 0.D0
C      HT(3) = ASBET
C      these are the az over el matrix values, WLL 11/00 corrected
C      AK(1) = - ASBET
C      AK(2) = ASALP * ACBET
C      AK(3) = ACALP * ACBET
C      VT(1) = 0.D0
C      VT(2) = ACALP
C      VT(3) = -ASALP
C      HT(1) = ACBET
C      HT(2) = ASALP * ASBET
C      HT(3) = ACALP * ASBET

C*****
C*****
C      GET COMPLEX ANTENNA GAIN FUNCTIONS
      GVTPA_A=vcomag(ifreq,j,i)
      GHTPA_A=vcrossmag(ifreq,j,i)
      GVTQA_A=hcrossmag(ifreq,j,i)

```

```

GHTQA_A=hcomag(ifreq,j,i)

PVTPA_P=vcophase(ifreq,j,i)
PHTPA_P=vcrossphase(ifreq,j,i)
PVTQA_P=hcrossphase(ifreq,j,i)
PHTQA_P=hcophase(ifreq,j,i)

CPVTPA = DCMPLX(DCOSD(PVTPA_P),DSIND(PVTPA_P))
CPHTPA = DCMPLX(DCOSD(PHTPA_P),DSIND(PHTPA_P))
CPVTQA = DCMPLX(DCOSD(PVTQA_P),DSIND(PVTQA_P))
CPHTQA = DCMPLX(DCOSD(PHTQA_P),DSIND(PHTQA_P))

C   FORM COMPLEX EFFECTIVE HEIGHT VECTORS
DO L=1,3
CEVPOL(L) = GVTPA_A*CPVTPA * VT(L) + GHTPA_A*CPHTPA * HT(L)
CEHPOL(L) = GVTQA_A*CPVTQA * VT(L) + GHTQA_A*CPHTQA * HT(L)
VE(L) = VT(L)
HE(L) = HT(L)
ENDDO

C   code to solve for CORTHO, a measure of the orthogonality
C
ctemp= DCMPLX(0.D0,0.D0)
DO L=1,3
ctemp=ctemp+CEVPOL(L)*DCONJG(CEHPOL(L))
ENDDO
CORTHO=CORTHO+ctemp*ACALP

C   FORM COEFS BIGA,BIGB,BIGC,BIGD
CBIGA = DCMPLX(0.D0,0.D0)
CBIGB = DCMPLX(0.D0,0.D0)
CBIGC = DCMPLX(0.D0,0.D0)
CBIGD = DCMPLX(0.D0,0.D0)
DO L=1,3
CBIGA = CBIGA + CEVPOL(L) * VE(L)
CBIGB = CBIGB + CEVPOL(L) * HE(L)
CBIGC = CBIGC + CEHPOL(L) * VE(L)
CBIGD = CBIGD + CEHPOL(L) * HE(L)
ENDDO

C   DO SUMS
C   these are for elevation over azimuth:
C   ANORM_V=ANORM_V+(CDABS(CBIGA)**2+CDABS(CBIGB)**2)*ACALP
C   ANORM_H=ANORM_H+(CDABS(CBIGC)**2+CDABS(CBIGD)**2)*ACALP
C   this is for azimuth over elevation:
ANORM_V=ANORM_V+(CDABS(CBIGA)**2+CDABS(CBIGB)**2)*ACBET
ANORM_H=ANORM_H+(CDABS(CBIGC)**2+CDABS(CBIGD)**2)*ACBET
ANORM_Vwl=ANORM_Vwl+(CDABS(GVTPA_A*CPVTPA)**2
1 +CDABS(GHTPA_A*CPHTPA)**2)
ANORM_Hwl=ANORM_Hwl+(CDABS(GVTQA_A*CPVTQA)**2
1 +CDABS(GHTQA_A*CPHTQA)**2)

ENDDO
ENDDO

C   WRITE(66,*)'do normalization with simple port sum (1) ?'
C   WRITE(66,*)' or using Gene Poe method w/ angle vectors (0) ?'
C   READ(55,*)inorm
C   IF(inorm.EQ.1) THEN
C   ANORM_V = DSQRT(ANORM_Vwl)
C   ANORM_H = DSQRT(ANORM_Hwl)
C   WRITE(*,*)' NORMALIZED ENERGIES (simple sum method):'
C   WRITE(*,*) ' +45,-45:',SNGL(ANORM_V),SNGL(ANORM_H)
C   WRITE(50,*)' NORMALIZED ENERGIES:'
C   WRITE(50,*) ' +45,-45:',SNGL(ANORM_V),SNGL(ANORM_H)
C   ELSE
ANORM_V = DSQRT(ANORM_V)
ANORM_H = DSQRT(ANORM_H)
CORTHO=CORTHO/(ANORM_V*ANORM_H)
WRITE(66,*)'CORTHO (orthogonality) = ',CORTHO

WRITE(50,*)' NORMALIZED ENERGIES:'

```

```

      WRITE(50,*) ' +45,-45:',SNGL(ANORM_V),SNGL(ANORM_H)
      WRITE(31,*) 'ifreqloop,SNGL(ANORM_V),SNGL(ANORM_H),' +45,-45:'
C      ENDIF

*****
C      COMPUTE COUPLING MATRIX OF STOKES PARAMETERS
*****
C      INITIALIZE SUMS TO ZERO
      ACOUP_VEVA = 0.D0
      ACOUP_HEVA = 0.D0
      ACOUP_UEVA = 0.D0
      ACOUP_4EVA = 0.D0

      ACOUP_VEHA = 0.D0
      ACOUP_HEHA = 0.D0
      ACOUP_UEHA = 0.D0
      ACOUP_4EHA = 0.D0

      ACOUP_VEUA_RT = 0.D0
      ACOUP_HEUA_RT = 0.D0
      ACOUP_VEUA_LT = 0.D0
      ACOUP_HEUA_LT = 0.D0
      ACOUP_VEUA = 0.D0
      ACOUP_HEUA = 0.D0
      ACOUP_UEUA = 0.D0
      ACOUP_4EUA = 0.D0

      ACOUP_VE4A_RT = 0.D0
      ACOUP_HE4A_RT = 0.D0
      ACOUP_VE4A_LT = 0.D0
      ACOUP_HE4A_LT = 0.D0
      ACOUP_VE4A = 0.D0
      ACOUP_HE4A = 0.D0
      ACOUP_UE4A = 0.D0
      ACOUP_4E4A = 0.D0

      DO j=1,numaz
C      BET  = AZRX(J)
C      ACBET = DCOSD(BET)
C      ASBET = DSIND(BET)

      DO I=numelcc-numeladd,numelcc+numeladd
C      DO I=1,numel
      BET  = aziangle(1,j,i)
      ACBET = DCOSD(BET)
      ASBET = DSIND(BET)
C      ALP  = ELR(I)
      ALP  = elevangle(1,j,i)
      ACALP = DCOSD(ALP)
      ASALP = DSIND(ALP)

C      DEFINE ANTENNA BORESIGHT AK
C      AK(1) = -ACALP * ASBET
C      AK(2) =  ASALP
C      AK(3) =  ACALP * ACBET
C      DEFINE ANTENNA RANGE TRANSMIT VERTICAL POL VT
C      VT(1) =  ASALP * ASBET
C      VT(2) =  ACALP
C      VT(3) = -ASALP * ACBET
C      DEFINE ANTENNA RANGE TRANSMIT HORIZONTAL POL HT
C      HT(1) =  ACBET
C      HT(2) =  0.D0
C      HT(3) =  ASBET
C      these are the az over el matrix values, WLL 11/00 corrected
      AK(1) = - ASBET
      AK(2) =  ASALP * ACBET
      AK(3) =  ACALP * ACBET
      VT(1) =  0.D0
      VT(2) =  ACALP
      VT(3) = -ASALP

```

```

HT(1) = ACBET
HT(2) = ASALP * ASBET
HT(3) = ACALP * ASBET

```

```

C*****
C*****

```

```

C GET COMPLEX ANTENNA GAIN FUNCTIONS

```

```

  GVTPA_A=vcomag(ifreq,j,i)
  GHTPA_A=vcrossmag(ifreq,j,i)
  GVTQA_A=hcrossmag(ifreq,j,i)
  GHTQA_A=hcomag(ifreq,j,i)

```

```

  PVTPA_P=vcophase(ifreq,j,i)
  PHTPA_P=vcrossphase(ifreq,j,i)
  PVTQA_P=hcrossphase(ifreq,j,i)
  PHTQA_P=hcophase(ifreq,j,i)

```

```

  CPVTPA = DCMPLX(DCOSD(PVTPA_P),DSIND(PVTPA_P))
  CPHTPA = DCMPLX(DCOSD(PHTPA_P),DSIND(PHTPA_P))
  CPVTQA = DCMPLX(DCOSD(PVTQA_P),DSIND(PVTQA_P))
  CPHTQA = DCMPLX(DCOSD(PHTQA_P),DSIND(PHTQA_P))

```

```

C FORM COMPLEX EFFECTIVE HEIGHT VECTORS

```

```

  DO L=1,3
  CEVPOLT(L) = GVTPA_A*CPVTPA * VT(L) + GHTPA_A*CPHTPA * HT(L)
  CEHPOLT(L) = GVTQA_A*CPVTQA * VT(L) + GHTQA_A*CPHTQA * HT(L)
  ENDDO

```

```

  IF(IANS_PRO.EQ.1)THEN

```

```

C GET LOCAL GEODETIC VE,HE VECTORS FOR VPOL
  CALL GEODETIC(AK,VE,HE)

```

```

C ROTATE COMPLEX EFFECTIVE HEIGHT VECTORS

```

```

C note: for no rotation, ACPOLROT =1 and ASPOLROT =0

```

```

  DO L=1,3
  CEVPOL(L) = ACPOLROT * CEVPOLT(L) + ASPOLROT * CEHPOLT(L)
  CEHPOL(L) = -ASPOLROT * CEVPOLT(L) + ACPOLROT * CEHPOLT(L)
  ENDDO

```

```

  ELSE

```

```

C WE USE ANTENNA RANGE POLARIZATION BASIS

```

```

  DO L=1,3
  VE(L) = VT(L)
  HE(L) = HT(L)
  CEVPOL(L) = CEVPOLT(L)
  CEHPOL(L) = CEHPOLT(L)
  ENDDO

```

```

  ENDIF

```

```

C FORM COEFS BIGA,BIGB,BIGC,BIGD

```

```

  CBIGA = DCMPLX(0.D0,0.D0)
  CBIGB = DCMPLX(0.D0,0.D0)
  CBIGC = DCMPLX(0.D0,0.D0)
  CBIGD = DCMPLX(0.D0,0.D0)
  DO L=1,3
  CBIGA = CBIGA + CEVPOL(L) * VE(L)
  CBIGB = CBIGB + CEVPOL(L) * HE(L)
  CBIGC = CBIGC + CEHPOL(L) * VE(L)
  CBIGD = CBIGD + CEHPOL(L) * HE(L)
  ENDDO

```

```

C NORMALIZE CBIGA,B,C,D

```

```

  CBIGA = CBIGA/ANORM_V
  CBIGB = CBIGB/ANORM_V
  CBIGC = CBIGC/ANORM_H
  CBIGD = CBIGD/ANORM_H

```

```

C COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO VA

```

```

GVEVA = 0.5D0 * CDABS(CBIGA + CBIGC)**2
GHEVA = 0.5D0 * CDABS(CBIGB + CBIGD)**2
GUEVA = 0.5D0 * DREAL((CBIGA+CBIGC)*DCONJG(CBIGB+CBIGD))
G4EVA = -0.5D0 * DIMAG((CBIGA+CBIGC)*DCONJG(CBIGB+CBIGD))

C   COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO HA
GVEHA = 0.5D0 * CDABS(CBIGA - CBIGC)**2
GHEHA = 0.5D0 * CDABS(CBIGB - CBIGD)**2
GUEHA = 0.5D0 * DREAL((CBIGA-CBIGC)*DCONJG(CBIGB-CBIGD))
G4EHA = -0.5D0 * DIMAG((CBIGA-CBIGC)*DCONJG(CBIGB-CBIGD))

C   COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO UA
GVEUA = CDABS(CBIGA)**2 - CDABS(CBIGC)**2
GHEUA = CDABS(CBIGB)**2 - CDABS(CBIGD)**2
GUEUA = DREAL(CBIGA*DCONJG(CBIGB) - CBIGC*DCONJG(CBIGD))
G4EUA = DIMAG(CBIGC*DCONJG(CBIGD) - CBIGA*DCONJG(CBIGB))

C   COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO 4A
GVE4A = -2.D0 * DIMAG( CBIGA * DCONJG(CBIGC) )
GHE4A = -2.D0 * DIMAG( CBIGB * DCONJG(CBIGD) )
GUE4A = -DIMAG(CBIGA*DCONJG(CBIGD) - CBIGC*DCONJG(CBIGB))
G4E4A = -DREAL(CBIGA*DCONJG(CBIGD) - CBIGC*DCONJG(CBIGB))

```

```

C*****
C   now sum up all the terms to get the Stokes matrix:
C   separate out the positive and negative terms as well
C   note, ACALP was changed to ACBET for az over el   WLL 12/00
C*****

```

```

ACOU_P_VEVA = ACOU_P_VEVA + GVEVA *ACBET
ACOU_P_HEVA = ACOU_P_HEVA + GHEVA *ACBET
ACOU_P_U EVA = ACOU_P_U EVA + GUEVA *ACBET
ACOU_P_4EVA = ACOU_P_4EVA + G4EVA *ACBET

```

```

ACOU_P_VEHA = ACOU_P_VEHA + GVEHA *ACBET
ACOU_P_HEHA = ACOU_P_HEHA + GHEHA *ACBET
ACOU_P_U EHA = ACOU_P_U EHA + GUEHA *ACBET
ACOU_P_4EHA = ACOU_P_4EHA + G4EHA *ACBET

```

```

ACOU_P_VEUA = ACOU_P_VEUA + GVEUA *ACBET
ACOU_P_HEUA = ACOU_P_HEUA + GHEUA *ACBET
IF(ASBET.GE.0.D0)THEN
ACOU_P_VEUA_RT = ACOU_P_VEUA_RT + GVEUA *ACBET
ACOU_P_HEUA_RT = ACOU_P_HEUA_RT + GHEUA *ACBET
ELSE
ACOU_P_VEUA_LT = ACOU_P_VEUA_LT + GVEUA *ACBET
ACOU_P_HEUA_LT = ACOU_P_HEUA_LT + GHEUA *ACBET
ENDIF
ACOU_P_U EUA = ACOU_P_U EUA + GUEUA *ACBET
ACOU_P_4EUA = ACOU_P_4EUA + G4EUA *ACBET

```

```

ACOU_P_VE4A = ACOU_P_VE4A + GVE4A *ACBET
ACOU_P_HE4A = ACOU_P_HE4A + GHE4A *ACBET
IF(ASBET.GE.0.D0)THEN
ACOU_P_VE4A_RT = ACOU_P_VE4A_RT + GVE4A *ACBET
ACOU_P_HE4A_RT = ACOU_P_HE4A_RT + GHE4A *ACBET
ELSE
ACOU_P_VE4A_LT = ACOU_P_VE4A_LT + GVE4A *ACBET
ACOU_P_HE4A_LT = ACOU_P_HE4A_LT + GHE4A *ACBET
ENDIF
ACOU_P_U E4A = ACOU_P_U E4A + GUE4A *ACBET
ACOU_P_4E4A = ACOU_P_4E4A + G4E4A *ACBET

```

```

ENDDO
ENDDO

```

```

C*****
C   a end of main do loop that finds the Stokes matrix
C*****
C*****

```

C*****

C write out the final data

C*****

```
WRITE(50,*) ' '
WRITE(50,*) ' STOKES COUPLING MATRIX : '
WRITE(50,*) ' VV HV UV 4V: '
WRITE(50,*) SNGL(ACOU_P_VEVA),SNGL(ACOU_HEVA),
+ SNGL(ACOU_U EVA),SNGL(ACOU_4EVA)
WRITE(29,*) SNGL(ACOU_P_VEVA),SNGL(ACOU_HEVA),
+ SNGL(ACOU_U EVA),SNGL(ACOU_4EVA)
WRITE(66,*) SNGL(ACOU_P_VEVA),SNGL(ACOU_HEVA),
+ SNGL(ACOU_U EVA),SNGL(ACOU_4EVA)

WRITE(50,*) ' '
WRITE(50,*) ' VH HH UH 4H: '
WRITE(50,*) SNGL(ACOU_P_VEHA),SNGL(ACOU_HEHA),
+ SNGL(ACOU_U EHA),SNGL(ACOU_4EHA)
WRITE(29,*) SNGL(ACOU_P_VEHA),SNGL(ACOU_HEHA),
+ SNGL(ACOU_U EHA),SNGL(ACOU_4EHA)
WRITE(66,*) SNGL(ACOU_P_VEHA),SNGL(ACOU_HEHA),
+ SNGL(ACOU_U EHA),SNGL(ACOU_4EHA)

WRITE(50,*) ' '
WRITE(50,*) ' VU HU UU 4U: '
WRITE(50,*) SNGL(ACOU_P_VEUA),SNGL(ACOU_HEUA),
+ SNGL(ACOU_U EUA),SNGL(ACOU_4EUA)
WRITE(29,*) SNGL(ACOU_P_VEUA),SNGL(ACOU_HEUA),
+ SNGL(ACOU_U EUA),SNGL(ACOU_4EUA)
WRITE(66,*) SNGL(ACOU_P_VEUA),SNGL(ACOU_HEUA),
+ SNGL(ACOU_U EUA),SNGL(ACOU_4EUA)

WRITE(50,*) ' VU HU RIGHT HEMISPHERE: '
WRITE(50,*) SNGL(ACOU_P_VEUA_RT),SNGL(ACOU_HEUA_RT)
WRITE(50,*) ' VU HU LEFT HEMISPHERE: '
WRITE(50,*) SNGL(ACOU_P_VEUA_LT),SNGL(ACOU_HEUA_LT)

WRITE(50,*) ' '
WRITE(50,*) ' V4 H4 U4 44: '
WRITE(50,*) SNGL(ACOU_P_VE4A),SNGL(ACOU_HE4A),
+ SNGL(ACOU_U E4A),SNGL(ACOU_4E4A)
WRITE(29,*) SNGL(ACOU_P_VE4A),SNGL(ACOU_HE4A),
+ SNGL(ACOU_U E4A),SNGL(ACOU_4E4A)
WRITE(66,*) SNGL(ACOU_P_VE4A),SNGL(ACOU_HE4A),
+ SNGL(ACOU_U E4A),SNGL(ACOU_4E4A)

WRITE(50,*) ' V4 H4 U4 44 RIGHT HEMISPHERE: '
WRITE(50,*) SNGL(ACOU_P_VE4A_RT),SNGL(ACOU_HE4A_RT)
WRITE(50,*) ' V4 H4 U4 44 LEFT HEMISPHERE: '
WRITE(50,*) SNGL(ACOU_P_VE4A_LT),SNGL(ACOU_HE4A_LT)
WRITE(29,*)
```

C*****

```
WRITE(66,*) ' rasterpost STOKES COUPLING MATRIX IN DB : '
TEDVU=SNGL(ACOU_P_VEVA)
TEDVU=10*LOG10(ABS(TEDVU))
TEDHU=SNGL(ACOU_P_VEHA)
TEDHU=10*LOG10(ABS(TEDHU))
TEDUU=SNGL(ACOU_P_U EVA)
TEDUU=10*LOG10(ABS(TEDUU))
TED4U=SNGL(ACOU_P_4EVA)
TED4U=10*LOG10(ABS(TED4U))
WRITE(66,*) SNGL(TEDVU),SNGL(TEDHU),SNGL(TEDUU),SNGL(TED4U)

TEDVU=SNGL(ACOU_P_VEHA)
TEDVU=10*LOG10(ABS(TEDVU))
TEDHU=SNGL(ACOU_P_VEHA)
TEDHU=10*LOG10(ABS(TEDHU))
TEDUU=SNGL(ACOU_P_U EVA)
```

```

TEDUU=10*LOG10(ABS(TEDUU))
TED4U=SNGL(ACOU_P_4EHA)
TED4U=10*LOG10(ABS(TED4U))
WRITE(66,*) SNGL(TEDVU),SNGL(TEDHU),SNGL(TEDUU),SNGL(TED4U)

TEDVU=SNGL(ACOU_P_VEUA)
TEDVU=10*LOG10(ABS(TEDVU))
TEDHU=SNGL(ACOU_P_HEUA)
TEDHU=10*LOG10(ABS(TEDHU))
TEDUU=SNGL(ACOU_P_UEUA)
TEDUU=10*LOG10(ABS(TEDUU))
TED4U=SNGL(ACOU_P_4EUA)
TED4U=10*LOG10(ABS(TED4U))
WRITE(66,*) SNGL(TEDVU),SNGL(TEDHU),SNGL(TEDUU),SNGL(TED4U)

TEDVU=SNGL(ACOU_P_VE4A)
TEDVU=10*LOG10(ABS(TEDVU))
TEDHU=SNGL(ACOU_P_HE4A)
TEDHU=10*LOG10(ABS(TEDHU))
TEDUU=SNGL(ACOU_P_UE4A)
TEDUU=10*LOG10(ABS(TEDUU))
TED4U=SNGL(ACOU_P_4E4A)
TED4U=10*LOG10(ABS(TED4U))
WRITE(66,*) SNGL(TEDVU),SNGL(TEDHU),SNGL(TEDUU),SNGL(TED4U)

WRITE(66,*)'rotation angle (deg. ) = ',SNGL(rotang)

C      loop back to the start
      IF(ifreqloop.LT.ifreqnum)GOTO 339
CLOSE(55)
CLOSE(66)
CLOSE(29)

C      compute centroid average
      acentroidavbet=0.D0
      acentroidavalp=0.D0
      DO kk=1,ifreqnum
        acentroidavbet=acentroidavbet-wBHTQ_BET(kk)-wBVTP_BET(kk)
        acentroidavalp=acentroidavalp-wBVTP_ALP(kk)-wBHTQ_ALP(kk)
      ENDDO
      acentroidavbet=acentroidavbet/ifreqnum/2.D0
      acentroidavalp=acentroidavalp/ifreqnum/2.D0
      WRITE(50,*)
      WRITE(36,*)
      WRITE(50,*)'average centroid values:'
      WRITE(36,*)'average centroid values:'
      WRITE(50,3345)acentroidavbet,acentroidavalp,
1    acentroidavbet+azencoder,acentroidavalp+elencoder
      WRITE(36,3345)acentroidavbet,acentroidavalp,
1    acentroidavbet+azencoder,acentroidavalp+elencoder

C
C
C      now convert to EIA and SCAN: *****
C      eiafind.for
C
C      this program calculates the eia and scan angles for a beam
C      given its azimuth and elevation angles relative to boresight
C      at thetao=45 deg.

C      REAL*8 AK(3),ZR(3),zx(3),zy(3),zm(3)

      WRITE(36,*)
C      switch back to Poe coordinate system
C      by taking negative of az and el centroid values
      BET  = -acentroidavbet
      ACBET = DCOSD(BET)
      ASBET = DSIND(BET)

      ALP  = -acentroidavalp

```



```

ACALP = DCOSD(ALP)
ASALP = DSIND(ALP)

C  DEFINE ANTENNA BORESIGHT AK
    azel=1.
    IF(azel.EQ.2.)THEN
C  these are for elevation over az
    AK(1) = -ACALP * ASBET
    AK(2) = ASALP
    AK(3) = ACALP * ACBET
    ELSE
C  these are for az over elevation
    AK(1) = - ASBET
    AK(2) = ACBET*ASALP
    AK(3) = ACALP * ACBET
    ENDIF

THETA0 = 45.D0
ZR(1) = 0.0
ZR(2) = DSIND(THETA0)
ZR(3) = -DCOSD(THETA0)

THETAant=DACOSD((ak(3))/SQRT((ak(1))**2+
1 (ak(2))**2+(ak(3))**2))

EIASAT=DACOSD(ZR(1)*AK(1)+ZR(2)*AK(2)+ZR(3)*AK(3))
C  WRITE(36,*)'eiasat, thetaant = ',eiasat,thetaant
C  earth radius in km
    r1=6378.14D0
    WRITE(36,*)'assuming earth radius, km =',r1
C  satellite height in km
    r2x=830.
    WRITE(36,*)'calculations assuming satellite ht, km =',r2x
    r2=r1+r2x
    eia=(DASIND(SIND(180-eiasat)/r1*r2))
C  WRITE(36,*)'ak', SNGL(AK(1)),SNGL(AK(2)),SNGL(AK(3))
    WRITE(36,*)'eia ',eia
C***** now find scan angle *****
C  first set x vector
    Zx(1) = 1.0D0
    Zx(2) = 0.0D0
    Zx(3) = 0.0D0
C  do crossproduct of z x x = y to find new y vector
    XP2=Zr(2)*zx(3)
    XP1=Zr(3)*zx(2)

    YP2=Zr(3)*zx(1)
    YP1=Zr(1)*zx(3)

    ZP2=Zr(1)*zx(2)
    ZP1=Zr(2)*zx(1)

    zy(1)=(XP2-XP1)
    zy(2)=(YP2-YP1)
    zy(3)=(ZP2-ZP1)
C  WRITE(6,*)'zx',SNGL(zx(1)),SNGL(zx(2)),SNGL(zx(3))
C  WRITE(6,*)'zy',SNGL(zy(1)),SNGL(zy(2)),SNGL(zy(3))
C  WRITE(6,*)'zr',SNGL(zr(1)),SNGL(zr(2)),SNGL(zr(3))
C  WRITE(6,*)'ak',SNGL(ak(1)),SNGL(ak(2)),SNGL(ak(3))
    zm(1)=ak(1)*zx(1)+ak(2)*zx(2)+ak(3)*zx(3)
    zm(2)=ak(1)*zy(1)+ak(2)*zy(2)+ak(3)*zy(3)
    zm(3)=ak(1)*zr(1)+ak(2)*zr(2)+ak(3)*zr(3)
C  WRITE(6,*)
C  WRITE(6,*)'zm ',SNGL(zm(1)),SNGL(zm(2)),SNGL(zm(3))
    THETAYVC=DACOSD((zm(3))/SQRT((zm(1))**2+
1 (zm(2))**2+(zm(3))**2))
    PHIYVC=DATAN2D((zm(2)),(zm(1)))
    PHIYVC=PHIYVC+90.D0

```

```

C      WRITE(36,*)'these should be the same, nadir angle'
C      WRITE(36,*)eiasat,THETAYVC
C      write(36,*)'scan angle ',PHIYVC

C *****
C      end of EIA and SCAN calculation *****
C *****

3345      FORMAT(15x,4(F13.4))

      GOTO 438
      OPEN(UNIT=21,FILE='array4v',STATUS='UNKNOWN')
      OPEN(UNIT=22,FILE='array4h',STATUS='UNKNOWN')
      OPEN(UNIT=23,FILE='array4vm',STATUS='UNKNOWN')
      OPEN(UNIT=24,FILE='array4hm',STATUS='UNKNOWN')
      I=1
      ax=1.0E-13
      DO k=numelcc-numeladd,numelcc+numeladd
C      DO k=1,numel
C      DO j=numaz,1,-1
C      WRITE(66,*)j,k,g4v(j,k)

      IF (j.EQ.1) THEN
      WRITE(21,2220) 10*LOG10(ABS(ax+g4v(j,numel-k+1)))
      WRITE(22,2220) 10*LOG10(ABS(ax+g4h(j,numel-k+1)))
      WRITE(23,2220) 1.E5*(ax+g4v(j,numel-k+1))
      WRITE(24,2220) 1.E5*(ax+g4h(j,numel-k+1))
      ELSEIF (j.EQ. numaz) THEN
      WRITE(21,2221) 10*LOG10(ABS(ax+g4v(j,numel-k+1)))
      WRITE(22,2221) 10*LOG10(ABS(ax+g4h(j,numel-k+1)))
      WRITE(23,2221) 1.E5*(ax+g4v(j,numel-k+1))
      WRITE(24,2221) 1.E5*(ax+g4h(j,numel-k+1))
      ELSE
      WRITE(21,2222) 10*LOG10(ABS(ax+g4v(j,numel-k+1)))
      WRITE(22,2222) 10*LOG10(ABS(ax+g4h(j,numel-k+1)))
      WRITE(23,2222) 1.E5*(ax+g4v(j,numel-k+1))
      WRITE(24,2222) 1.E5*(ax+g4h(j,numel-k+1))
      END IF
      ENDDO
      ENDDO
      WRITE(66,*)'4th stokes in dB for terms ',jaz,kel
      WRITE(66,*)SNGL(10*LOG10(ABS(ax+g4v(jaz,kel))))
      WRITE(66,*)SNGL(10*LOG10(ABS(ax+g4h(jaz,kel))))
      WRITE(66,*)'enter az and el integer positions for printout'
      READ(55,*)jaz,kel
      WRITE(66,*)SNGL(10*LOG10(ABS(ax+g4v(jaz,kel))))
      WRITE(66,*)SNGL(10*LOG10(ABS(ax+g4h(jaz,kel))))
      WRITE(66,*)'enter az and el integer positions for printout'
      READ(55,*)jaz,kel
      WRITE(66,*)SNGL(10*LOG10(ABS(ax+g4v(jaz,kel))))
      WRITE(66,*)SNGL(10*LOG10(ABS(ax+g4h(jaz,kel))))
      WRITE(66,*)'enter az and el integer positions for printout'
      READ(55,*)jaz,kel
      WRITE(66,*)SNGL(10*LOG10(ABS(ax+g4v(jaz,kel))))
      WRITE(66,*)SNGL(10*LOG10(ABS(ax+g4h(jaz,kel))))
      WRITE(66,*)'enter az and el integer positions for printout'
      READ(55,*)jaz,kel
      WRITE(66,*)SNGL(10*LOG10(ABS(ax+g4v(jaz,kel))))
      WRITE(66,*)SNGL(10*LOG10(ABS(ax+g4h(jaz,kel))))
      CLOSE(21)
      CLOSE(22)
      CLOSE(23)
      CLOSE(24)
438      CONTINUE

2220 FORMAT (2X,F15.1)
2221 FORMAT (F15.1,$)

```

2222 FORMAT (2X,F15.1,\$)

STOP
END

C*****
C*****

C*****

SUBROUTINE GEODETIC(AK,VE,HE)

C
C COMPUTES THE LOCAL GEODETIC VERTICAL, HORIZONTAL POLARIZATIONS
C FROM PROPAGATION VECTOR

IMPLICIT REAL*8(A-H,O-Z)

DIMENSION AK(3),VE(3),HE(3),ZR(3)
DATA IFLAG/0/

C wendy took out this IF loop
C IF(IFLAG.EQ.0)THEN
C IFLAG = 1
C THETA0 = 45.D0
C WE DEFINE LOCAL GEODETIC VERTICAL AND HORIZONTAL POLARIZATIONS
C THETA0 IS SSMI HALF CONE ANGLE
C SPACECRAFT ZENITH ZEN(I),I=1,3:
C ZEN = -COS(THETA0)*YHATP + SIN(THETA0)*ZHATP
C ZR(1) = 0.0
C ZR(2) = DSIND(THETA0)
C ZR(3) = -DCOSD(THETA0)
C ENDIF

C COMPUTE LOCAL GEODETIC NORMAL

HE(1) = AK(2)*ZR(3) - AK(3)*ZR(2)
HE(2) = AK(3)*ZR(1) - AK(1)*ZR(3)
HE(3) = AK(1)*ZR(2) - AK(2)*ZR(1)
ANORM = 0.D0
DO I=1,3
ANORM = ANORM + HE(I)**2
ENDDO
ANORM = DSQRT(ANORM)

DO I=1,3
HE(I) = -HE(I)/ANORM
ENDDO

VE(1) = AK(2)*HE(3) - AK(3)*HE(2)
VE(2) = AK(3)*HE(1) - AK(1)*HE(3)
VE(3) = AK(1)*HE(2) - AK(2)*HE(1)

RETURN
END

C*****
C*****
C*****

SUBROUTINE PURDYTRANSFORM(azianglez,elevanglez,
1 azianglenew,elevanglenew,freq)

C program to convert the raw range az/el values to the
C spacecraft values using Bill Purdy's coordinate
C transformation

IMPLICIT REAL*8 (A-H,O-Z)
REAL*8 XIN, YIN, ZIN
REAL*8 FOCLN, DIAM, OFFSET
REAL*8 THETA1, THETA2, ANGLECF
REAL*8 A,B,C, YI, YII
REAL*8 zx(3),zy(3)

```

REAL*8 THELEV, THAZIM, DIST, FLAG
REAL*8 X1,X2,Y1,Y2,Z1,Z2,RPD,DPR
REAL*8 THETAZVC, PHIZVC, THETAZVC, PHIXVC
REAL*8 XVC(3), YVC(3), ZVC(3),VCW(3,3),ZVCW(3)
REAL*8 THZ,PHZ,THX,PHX
  REAL*8 XS(3),YS(3),ZS(3)
  REAL*8 XX(3),YY(3),ZZ(3)
  REAL*8 XM(3),YM(3),ZM(3)
  REAL*8 XMm(3),YmM(3),ZmM(3),ZMM2(3)
  REAL*8 Xr(3),Yr(3),Zr(3),zmm3(3),ZMM4(3),yp(3)
REAL*8 AK(3),HT(3),VT(3)

```

```

REAL*8 X,Y,Z,THETAZ,PHIIN,R
REAL*8 DOT,DZX,VC(3,3)
REAL*8 XNEW,YNEW,ZNEW

```

```

RPD=3.14159265/180.
DPR=180./3.14159265

```

```

C*****
C*****
C first input the range coordinates, in az over el coordinate system
C*****

```

```

C note, use the range coordinate system:
C also same as Tiera convention
C az off to the left , elevation is down
C right hand rule

```

```

  Aziangle=azianglez
  Elevangle=elevanglez
  IF(freq.GT.21.)THEN
    buckline=70.9486D0
  ENDIF

```

```

  GOTO 3355
  IF(freq.GT.8.)THEN
    buckline=71.2482D0
  ENDIF

```

```

  GOTO 3355
  IF(freq.GT.5.)THEN
    buckline=71.514D0
  ENDIF

```

```

3355 CONTINUE

```

```

C*****
C first need to convert to theta/phi using az over el transform:
C reverse signs of az and elevation first, since code is written
C with left hand rule.
  BET = -aziangle
  ACBET = DCOSD(BET)
  ASBET = DSIND(BET)

```

```

  ALP = -elevangle
  ACALP = DCOSD(ALP)
  ASALP = DSIND(ALP)

```

```

C DEFINE ANTENNA BORESIGHT AK, el over az
C AK(1) = -ACALP * ASBET
C AK(2) = ASALP
C AK(3) = ACALP * ACBET
C DEFINE ANTENNA RANGE TRANSMIT VERT. POL VT, el over az
C VT(1) = ASALP * ASBET
C VT(2) = ACALP
C VT(3) = -ASALP * ACBET
C DEFINE ANTENNA RANGE TRANSMIT HORIZ. POL HT, el over az
C HT(1) = ACBET
C HT(2) = 0.D0
C HT(3) = ASBET
C these are the az over el matrix values, WLL 11/00 corrected

```

```

zz(1) = - ASBET
zz(2) = ASALP * ACBET
zz(3) = ACALP * ACBET
WRITE(6,*)'zz() ',zz(1),zz(2),zz(3)
yy(1) = 0.D0
yy(2) = ACALP
yy(3) = -ASALP
xx(1) = ACBET
xx(2) = ASALP * ASBET
xx(3) = ACALP * ASBET
C*****
C*****
C first transform to main range coord. system
C*****

C***set up x-axis of range from theta/phi information*****
THETAXVC=90.D0-buckline
PHIXVC=-90.D0
XS(1)=DSIND(THETAXVC)*DCOSD(PHIXVC)
XS(2)=DSIND(THETAXVC)*DSIND(PHIXVC)
XS(3)=DCOSD(THETAXVC)
C***set up y-axis of range from theta/phi information*****
THETAYVC=90.D0
PHIYVC=0.D0
YS(1)=DSIND(THETAYVC)*DCOSD(PHIYVC)
YS(2)=DSIND(THETAYVC)*DSIND(PHIYVC)
YS(3)=DCOSD(THETAYVC)
C***set up z-axis of range from theta/phi information*****
THETAZVC=buckline
PHIZVC=90.D0
ZS(1)=DSIND(THETAZVC)*DCOSD(PHIZVC)
ZS(2)=DSIND(THETAZVC)*DSIND(PHIZVC)
ZS(3)=DCOSD(THETAZVC)
zmm(1)=ZZ(1)*XS(1)+ZZ(2)*XS(2)+ZZ(3)*XS(3)
zmm(2)=ZZ(1)*YS(1)+ZZ(2)*YS(2)+ZZ(3)*YS(3)
zmm(3)=ZZ(1)*ZS(1)+ZZ(2)*ZS(2)+ZZ(3)*ZS(3)

C need to invert the matrix to reverse the transform direction
C for orthogonal matrices, the inverse is the transpose
C

C WRITE(6,*)
C WRITE(6,*)'zmm(1) ',zmm(1)
C WRITE(6,*)'zmm(2) ',zmm(2)
C WRITE(6,*)'zmm(3) ',zmm(3)
THETAZVC=DACOSD((zmm(3))/SQRT((zmm(1))**2+
1 (zmm(2))**2+(zmm(3))**2))
PHIZVC=DATAN2D((zmm(2)),(zmm(1)))
C WRITE(6,*)'theta/phi z-axis ',SNGL(thetazvc),SNGL(phizvc)
zz(1)=zmm(1)
zz(2)=zmm(2)
zz(3)=zmm(3)
C*****
C*****
C*****
C these are the main spacecraft coordinates
C these are X cs-rot, Y cs-rot, and Z cs-rot
C get these from Bill Purdy
C need to add code to make these axes orthogonal
C*****
C***set up x-axis of range from theta/phi information*****
THETAXVC=90.1656D0
PHIXVC=90.0361D0
XS(1)=DSIND(THETAXVC)*DCOSD(PHIXVC)
XS(2)=DSIND(THETAXVC)*DSIND(PHIXVC)
XS(3)=DCOSD(THETAXVC)
C***set up y-axis of range from theta/phi information*****
THETAYVC=153.9843D0
PHIYVC=180.3753D0

```

```

      YS(1)=DSIND(THETAYVC)*DCOSD(PHIYVC)
      YS(2)=DSIND(THETAYVC)*DSIND(PHIYVC)
      YS(3)=DCOSD(THETAYVC)
C***set up z-axis of range from theta/phi information*****
      THETAZVC=63.9849D0
      PHIZVC=179.9553D0
      ZS(1)=DSIND(THETAZVC)*DCOSD(PHIZVC)
      ZS(2)=DSIND(THETAZVC)*DSIND(PHIZVC)
      ZS(3)=DCOSD(THETAZVC)

C*****
C*****
C*****
C   now convert to cs-rot (spacecraft) coordinate system
C   now do the matrix multiplication to get transformed coordinates
C*****

C   this is the main matrix to transform
C   xmm(1)=XVC(1)*XS(1)+XVC(2)*YS(1)+XVC(3)*ZS(1)
C   xmm(2)=XVC(1)*XS(2)+XVC(2)*YS(2)+XVC(3)*ZS(2)
C   xmm(3)=XVC(1)*XS(3)+XVC(2)*YS(3)+XVC(3)*ZS(3)

C   ymm(1)=YVC(1)*XS(1)+YVC(2)*YS(1)+YVC(3)*ZS(1)
C   ymm(2)=YVC(1)*XS(2)+YVC(2)*YS(2)+YVC(3)*ZS(2)
C   ymm(3)=YVC(1)*XS(3)+YVC(2)*YS(3)+YVC(3)*ZS(3)

C   zmm(1)=ZVC(1)*XS(1)+ZVC(2)*YS(1)+ZVC(3)*ZS(1)
C   zmm(2)=ZVC(1)*XS(2)+ZVC(2)*YS(2)+ZVC(3)*ZS(2)
C   zmm(3)=ZVC(1)*XS(3)+ZVC(2)*YS(3)+ZVC(3)*ZS(3)

C   this transforms the point in the old coord. system
C   to the new coordinate system.
C   note: there are four versions of this kind of transformation

      zmm(1)=ZZ(1)*XS(1)+ZZ(2)*XS(2)+ZZ(3)*XS(3)
      zmm(2)=ZZ(1)*YS(1)+ZZ(2)*YS(2)+ZZ(3)*YS(3)
      zmm(3)=ZZ(1)*ZS(1)+ZZ(2)*ZS(2)+ZZ(3)*ZS(3)

C   need to invert the matrix to reverse the transform direction
C   for orthogonal matrices, the inverse is the transpose
C

C   WRITE(6,*)
C   WRITE(6,*)'zmm(1)',zmm(1)
C   WRITE(6,*)'zmm(2)',zmm(2)
C   WRITE(6,*)'zmm(3)',zmm(3)
      THETAZVC=DACOSD((zmm(3))/SQRT((zmm(1))**2+
1 (zmm(2))**2+(zmm(3))**2))
      PHIZVC=DATAN2D((zmm(2)),(zmm(1)))
C   WRITE(6,*)'theta/phi z-axis ',SNGL(thetazvc),SNGL(phizvc)
C   WRITE(6,*)

C*****
C   now convert from spacecraft to Poe coord system
C*****
C***set up x-axis of range from theta/phi information*****
C   nominal:      THETAXVC=90.D0
C   nominal:      PHIXVC=90.D0
      THETAXVC=90.0D0
      PHIXVC=0.0D0
      XS(1)=DSIND(THETAXVC)*DCOSD(PHIXVC)
      XS(2)=DSIND(THETAXVC)*DSIND(PHIXVC)
      XS(3)=DCOSD(THETAXVC)
C***set up y-axis of range from theta/phi information*****
C   nominal:      THETAYVC=90.D0
C   nominal:      PHIYVC=90.D0
      THETAYVC=45.D0
      PHIYVC=-90.0D0
      YS(1)=DSIND(THETAYVC)*DCOSD(PHIYVC)

```

```

      YS(2)=DSIND(THETAYVC)*DSIND(PHIYVC)
      YS(3)=DCOSD(THETAYVC)
C****set up z-axis of range from theta/phi information*****
C nominal:      THETAZVC=0.D0
C nominal:      PHIZVC=0.D0
      THETAZVC=135.D0
      PHIZVC=-90.D0
      ZS(1)=DSIND(THETAZVC)*DCOSD(PHIZVC)
      ZS(2)=DSIND(THETAZVC)*DSIND(PHIZVC)
      ZS(3)=DCOSD(THETAZVC)

C now do coord. transformation again
      zmm2(1)=Zmm(1)*XS(1)+Zmm(2)*XS(2)+Zmm(3)*XS(3)
      zmm2(2)=Zmm(1)*YS(1)+Zmm(2)*YS(2)+Zmm(3)*YS(3)
      zmm2(3)=Zmm(1)*ZS(1)+Zmm(2)*ZS(2)+Zmm(3)*ZS(3)

C      WRITE(6,*)
C      WRITE(6,*)'zmm2(1) ',zmm2(1)
C      WRITE(6,*)'zmm2(2) ',zmm2(2)
C      WRITE(6,*)'zmm2(3) ',zmm2(3)
      THETAZVC=DACOSD((zmm2(3))/SQRT((zmm2(1))**2+
1 (zmm2(2))**2+(zmm2(3))**2))
      PHIZVC=DATAN2D((zmm2(2)),(zmm2(1)))
C      WRITE(6,*)'theta/phi z-axis ',SNGL(thetazvc),SNGL(phizvc)
C      WRITE(6,*)

C now convert back to az over el system
      azianglenew=-DASIND(-zmm2(1))
      BET = -azianglenew
      ACBET = DCOSD(BET)
      elevanglenew=-DASIND(zmm2(2)/ACBET)
C      WRITE(6,*)'buckline,freq
C      WRITE(6,*)'aziangle old/new, elev. Angle old/new'
C      WRITE(6,*)SNGL(azianglez),SNGL(azianglenew),
C      1 SNGL(elevanglez),SNGL(elevanglenew)
C      WRITE(6,*)'hit return'
C      READ(5,*)
      RETURN
      END
C*****
C*****
C*****
      SUBROUTINE RAST(AK,ALP,BET,VT,HT)

C      UPDATED 5/11/99 BY GENE POE
C      MAKES THE ELEVATION ROTATION CCW ABOUT X-AXIS
C
C      THIS ROUTINE COMPUTES ALP,BET AND VT,HT POLARIZATIONS OF
C      THE TRANSMIT ANTENNA IN THE WINDSAT COORD SYSTEM

C      INPUT:
C      AK(I) I=1,...,3 (PROPAGATION VECTOR IN XPP,YPP,ZPP SYSTEM)
C      OUTPUT:
C      ALP,BET (D) ARE THE EL,AZ ROTATIONS OF ANTENNA RANGE
C      VT(I) I=1,...,3 (VERTICAL POL OF SOURCE IN WINDSAT COORD)
C      HT(I) I=1,...,3 (HOR POL OF SOURCE IN WINDSAT COORD)
C
C      THE RASTER IS SET UP TO TILT THE ANTENNA BORESIGHT ABOUT THE X-AXIS
C      ANGLE ALP CCW AND THEN ROTATE THE ABOUT THE NEW YP CCW AXIS BY
C      ANGLE BETA (THIS IS ALSO CALLED AZ/EL SCAN GEOMETRY)
C
C      THE TRANSMIT ANTENNA IS DIRECTED TOWARDS -Z WITH VERTICAL POL
C      ALONG Y AND HORIZONTAL POL ALONG X WHERE WE HAVE A RIGHTHAND COORD
C      SYSTEM.

C      EQUIVALENTLY THE SOURCE BORESIGHT, V-PORT AND H-PORT MAY BE EXPRESSED
C      IN TERMS OF THE WINDSAT ANTENNA COORDINATES:
C      the following are for elevation over azimuth positioner: (wrong for our system)
C      note: matrix formed by multiplying [az] * [el] rotation matrices

```

```

C   V POL (SOURCE) = [ SIN(ALP)SIN(BET), COS(ALP),-SIN(ALP)COS(BET)]
C   H POL (SOURCE) = [      COS(BET), 0      , SIN(BET)]
C   KT  (SOURCE) = [ COS(ALP)SIN(BET),-SIN(ALP),-COS(ALP)COS(BET)]
C   ANTENNA BORESIGHT IS ZR = -KT
C   ALP is elevation
C   BET is azimuth
C   this is the correct matrix for az over el positioner:
C   modified by WLL on 11/00
C   note: matrix formed by multiplying [el] * [az] rotation matrices
C   V POL (SOURCE) = [      0      , COS(ALP), -SIN(ALP)]
C   H POL (SOURCE) = [ COS(BET), SIN(ALP)SIN(BET) , COS(ALP)SIN(BET)]
C   ZR  (SOURCE) = [ -SIN(BET),SIN(ALP)COS(BET),COS(ALP)COS(BET)]
C
C   ON FIRST CALL WE INITIALIZE ANTENNA PATTERN PARAMETERS
C   SUBSEQUENT CALLS WE RETURN ALP AND BET FOR INPUT THETA,PHI

      IMPLICIT REAL*8(A-H,O-Z)

      DIMENSION AK(3),VT(3),HT(3)

C   COMPUTE ALP AND BET

C   old values from el over az, incorrect:
C   ALP = DASIND(AK(2))
C   BET = DATAND(-AK(1)/AK(3))
C   good values for az over el positioner:
      BET = DASIND(-AK(1))
      ALP = DATAND(AK(2)/AK(3))

C   COMPUTE RASTER POLARIZATIONS VT AND HT
      ASALP = SIND(ALP)
      ACALP = COSD(ALP)
      ASBET = SIND(BET)
      ACBET = COSD(BET)

      VT(1) = 0.D0
      VT(2) = ACALP
      VT(3) = -ASALP
      HT(1) = ACBET
      HT(2) = ASALP * ASBET
      HT(3) = ACALP * ASBET

      RETURN
      END
C*****
C*****
C*****

```


Appendix E.

POE Code, excerpts for CP and V/H Polarization

Rastervhnew.for (excerpts):

```

C   DEFINE BORESIGHT AND VT- AND HT-POL UNIT VECTORS
C   FOR V-POL
      DO L=1,3
        AK0VPOL(L) = AK(L)
        AV0VPOL(L) = VT(L)
        AH0VPOL(L) = HT(L)
      ENDDO
      .....
      .....
      .....

C   save centroid information so can print out average later
      wBHTQ_BET(ifreqloop)=BHTQ_BET
      wBHTQ_ALP(ifreqloop)=BHTQ_ALP
      wBVTP_BET(ifreqloop)=BVTP_BET
      wBVTP_ALP(ifreqloop)=BVTP_ALP

      WRITE(50,3341)freqprint,SNGL(-BHTQ_BET),SNGL(-BHTQ_ALP),
1  SNGL(-BHTQ_BET+azencoder),SNGL(-BHTQ_ALP+elencoder)

      WRITE(36,3341)freqprint,SNGL(-BHTQ_BET),SNGL(-BHTQ_ALP),
1  SNGL(-BHTQ_BET+azencoder),SNGL(-BHTQ_ALP+elencoder)
3341  FORMAT(7H-45-POL,2x,F6.3,4(F13.4))

C*****
C   COMPLETED CENTROID COMPUTATIONS
C*****

C   WRITE(*,*) ' TO PROJECT ANTENNA BEAMS ONTO EARTH, ENTER 1:'
C   WRITE(*,*) '          ANTENNA BEAMS ONTO ANT RANGE,  2:'
C   WRITE(*,*) ' ENTER SELECTION:'
C   READ(55,*) IANS_PRO
      IANS_PRO=1
      POLROT = 0.D0
      ACPOLROT = DCOSD(POLROT)
      ASPOLROT = DSIND(POLROT)

      IF(IANS_PRO.EQ.1)THEN
C   COMPUTE POLARIZATION ROTATION OF BEAM WRT EARTH POLARIZATION
      WRITE(50,*) ' WE PROJECT ANTENNA BEAMS ONTO EARTH SURFACE:'

      CALL GEODETIC(AK0VPOL,VE,HE)
      DO L=1,3
        PE(L) =( VE(L) + HE(L) )/ROOT2
        AME(L)=( VE(L) - HE(L) )/ROOT2
      ENDDO

      ACPOLROTV = 0.D0
      ACPOLROTH = 0.D0
      DO L=1,3
        ACPOLROTV = ACPOLROTV + AV0VPOL(L)*PE(L)
        ACPOLROTH = ACPOLROTH + AV0VPOL(L)*AME(L)
      ENDDO

      POLROTV  = DATAN2D(ACPOLROTH,ACPOLROTV)
      WRITE(50,*) ' POL ROT FOR + 45 POL:',SNGL(POLROTV)

      CALL GEODETIC(AK0HPOL,VE,HE)

      ACPOLROTV = 0.D0
      ACPOLROTH = 0.D0
      DO L=1,3
        ACPOLROTV = ACPOLROTV + AV0HPOL(L)*PE(L)

```

```

      ACPOLROTH = ACPOLROTH + AV0HPOL(L)*AME(L)
      ENDDO

      POLROTH = DATAN2D(ACPOLROTH,ACPOLROTV)
      WRITE(50,*) ' POL ROT FOR -45 POL:',SNGL(POLROTH)

C   WRITE(*,*) ' WANT TO ROTATE FEEDHORN TO ALIGN THE EARTH AND '
C   WRITE(*,*) ' ANTENNA POLARIZATION BASIS? (Y/N):'
C   READ(55,(A)) ANS_ROT
      ANS_ROT='Y'
      IF(ANS_ROT.EQ.'Y'.OR.ANS_ROT.EQ.'y')THEN
C   WRITE(*,*) ' WANT TO USE AVERAGE OF ABOVE ROTATION ANGLE?(Y/N):'
C   READ(55,(A)) ANS_DUM
      ANS_DUM='n'

      IF(ANS_DUM.EQ.'Y'.OR.ANS_DUM.EQ.'y')THEN
        POLROT = 0.5D0*(POLROTV + POLROTH)
      ELSE
C   WRITE(*,*) ' Input Pol Offset in deg:'
      POLROT=0.D0
      ENDIF

      WRITE(50,*) ' WE ROTATE FEEDHORN (D):',SNGL(POLROT)
      ACPOLROT = DCOSD(POLROT)
      ASPOLROT = DSIND(POLROT)

      ELSE
      WRITE(50,*) ' WE DO NOT ROTATE FEEDHORN TO ALIGN EARTH AND '
      WRITE(50,*) ' ANTENNA RANGE POLARIZATION BASIS:'

      ENDIF
      ENDIF

C
C*****
C   NEXT WE FIND THE NORMALIZATION FACTORS FOR THE ANTENNA
C   BEAM ENERGIES

      ANORM_V = 0.D0
      ANORM_H = 0.D0
      CORTH0 = DCMPLX(0.D0,0.D0)

      DO j=1,numaz
C   BET = AZRX(J)
C   ACBET = DCOSD(BET)
C   ASBET = DSIND(BET)

      DO I=numelcc-numeladd,numelcc+numeladd
C   DO I=1,numel
      BET = aziangle(1,j,i)
      ACBET = DCOSD(BET)
      ASBET = DSIND(BET)
C   ALP = ELR(I)
      ALP = elevangle(1,j,i)
      ACALP = DCOSD(ALP)
      ASALP = DSIND(ALP)

C   DEFINE ANTENNA BORESIGHT AK
C   AK(1) = -ACALP * ASBET
C   AK(2) = ASALP
C   AK(3) = ACALP * ACBET
C   DEFINE ANTENNA RANGE TRANSMIT VERTICAL POL VT
C   VT(1) = ASALP * ASBET
C   VT(2) = ACALP
C   VT(3) = -ASALP * ACBET
C   DEFINE ANTENNA RANGE TRANSMIT HORIZONTAL POL HT
C   HT(1) = ACBET
C   HT(2) = 0.D0
C   HT(3) = ASBET
C   these are the az over el matrix values, WLL 11/00 corrected
      AK(1) = - ASBET

```

```

      AK(2) = ASALP * ACBET
      AK(3) = ACALP * ACBET
      VT(1) = 0.D0
      VT(2) = ACALP
      VT(3) = -ASALP
      HT(1) = ACBET
      HT(2) = ASALP * ASBET
      HT(3) = ACALP * ASBET

C*****
C*****
C   GET COMPLEX ANTENNA GAIN FUNCTIONS
      GVTPA_A=vcomag(ifreq,j,i)
      GHTPA_A=vcrossmag(ifreq,j,i)
      GVTQA_A=hcrossmag(ifreq,j,i)
      GHTQA_A=hcomag(ifreq,j,i)

      PVTPA_P=vcophase(ifreq,j,i)
      PHTPA_P=vcrossphase(ifreq,j,i)
      PVTQA_P=hcrossphase(ifreq,j,i)
      PHTQA_P=hcophase(ifreq,j,i)

      CPVTPA = DCMLPX(DCOSD(PVTPA_P),DSIND(PVTPA_P))
      CPHTPA = DCMLPX(DCOSD(PHTPA_P),DSIND(PHTPA_P))
      CPVTQA = DCMLPX(DCOSD(PVTQA_P),DSIND(PVTQA_P))
      CPHTQA = DCMLPX(DCOSD(PHTQA_P),DSIND(PHTQA_P))

C   FORM COMPLEX EFFECTIVE HEIGHT VECTORS
      DO L=1,3
      CEVPOL(L) = GVTPA_A*CPVTPA * VT(L) + GHTPA_A*CPHTPA * HT(L)
      CEHPOL(L) = GVTQA_A*CPVTQA * VT(L) + GHTQA_A*CPHTQA * HT(L)
      VE(L) = VT(L)
      HE(L) = HT(L)
      ENDDO

C   code to solve for CORTHO, a measure of the orthogonality
C
      ctemp= DCMLPX(0.D0,0.D0)
      DO L=1,3
      ctemp=ctemp+CEVPOL(L)*DCONJG(CEHPOL(L))
      ENDDO
      CORTHO=CORTHO+ctemp*ACALP
C   FORM COEFS BIGA,BIGB,BIGC,BIGD
      CBIGA = DCMLPX(0.D0,0.D0)
      CBIGB = DCMLPX(0.D0,0.D0)
      CBIGC = DCMLPX(0.D0,0.D0)
      CBIGD = DCMLPX(0.D0,0.D0)
      DO L=1,3
      CBIGA = CBIGA + CEVPOL(L) * VE(L)
      CBIGB = CBIGB + CEVPOL(L) * HE(L)
      CBIGC = CBIGC + CEHPOL(L) * VE(L)
      CBIGD = CBIGD + CEHPOL(L) * HE(L)
      ENDDO

C   DO SUMS
C   these are for elevation over azimuth:
C   ANORM_V=ANORM_V+(CDABS(CBIGA)**2+CDABS(CBIGB)**2)*ACALP
C   ANORM_H=ANORM_H+(CDABS(CBIGC)**2+CDABS(CBIGD)**2)*ACALP
C   this is for azimuth over elevation:
      ANORM_V=ANORM_V+(CDABS(CBIGA)**2+CDABS(CBIGB)**2)*ACBET
      ANORM_H=ANORM_H+(CDABS(CBIGC)**2+CDABS(CBIGD)**2)*ACBET
      ANORM_Vwl=ANORM_Vwl+(CDABS(GVTPA_A*CPVTPA)**2
1 +CDABS(GHTPA_A*CPHTPA)**2)
      ANORM_Hwl=ANORM_Hwl+(CDABS(GVTQA_A*CPVTQA)**2
1 +CDABS(GHTQA_A*CPHTQA)**2)

      ENDDO
      ENDDO

C   WRITE(66,*)'do normalization with simple port sum (1) ?'
C   WRITE(66,*)' or using Gene Poe method w/ angle vectors (0) ?'
C   READ(55,*)inorm

```

```

C      IF(inorm.EQ.1) THEN
C      ANORM_V = DSQRT(ANORM_Vwl)
C      ANORM_H = DSQRT(ANORM_Hwl)
C      WRITE(*,*)' NORMALIZED ENERGIES (simple sum method):'
C      WRITE(*,*) ' +45,-45:',SNGL(ANORM_V),SNGL(ANORM_H)
C      WRITE(50,*)' NORMALIZED ENERGIES:'
C      WRITE(50,*) ' +45,-45:',SNGL(ANORM_V),SNGL(ANORM_H)
C      ELSE
C      ANORM_V = DSQRT(ANORM_V)
C      ANORM_H = DSQRT(ANORM_H)
C      CORTHO=CORTHO/(ANORM_V*ANORM_H)
C      WRITE(66,*)'CORTHO (orthogonality) = ',CORTHO

C      WRITE(50,*)' NORMALIZED ENERGIES:'
C      WRITE(50,*) ' +45,-45:',SNGL(ANORM_V),SNGL(ANORM_H)
C      WRITE(31,*) ifreqloop,SNGL(ANORM_V),SNGL(ANORM_H),' +45,-45:'
C      ENDIF

```

```

*****

```

```

C      COMPUTE COUPLING MATRIX OF STOKES PARAMETERS

```

```

*****

```

```

C      INITIALIZE SUMS TO ZERO

```

```

      ACoup_VEVA = 0.D0
      ACoup_HEVA = 0.D0
      ACoup_UEVA = 0.D0
      ACoup_4EVA = 0.D0

```

```

      ACoup_VEHA = 0.D0
      ACoup_HEHA = 0.D0
      ACoup_UEHA = 0.D0
      ACoup_4EHA = 0.D0

```

```

      ACoup_VEUA_RT = 0.D0
      ACoup_HEUA_RT = 0.D0
      ACoup_VEUA_LT = 0.D0
      ACoup_HEUA_LT = 0.D0
      ACoup_VEUA=0.D0
      ACoup_HEUA=0.D0
      ACoup_UEUA = 0.D0
      ACoup_4EUA = 0.D0

```

```

      ACoup_VE4A_RT = 0.D0
      ACoup_HE4A_RT = 0.D0
      ACoup_VE4A_LT = 0.D0
      ACoup_HE4A_LT = 0.D0
      ACoup_VE4A=0.D0
      ACoup_HE4A=0.D0
      ACoup_UE4A = 0.D0
      ACoup_4E4A = 0.D0

```

```

      DO j=1,numaz

```

```

C      BET  = AZRX(J)
C      ACBET = DCOSD(BET)
C      ASBET = DSIND(BET)

```

```

      DO I=numelcc-numeladd,numelcc+numeladd

```

```

C      DO I=1,numel
C      BET  = aziangle(1,j,i)
C      ACBET = DCOSD(BET)
C      ASBET = DSIND(BET)

```

```

C      ALP  = ELR(I)
C      ALP  = elevangle(1,j,i)
C      ACALP = DCOSD(ALP)
C      ASALP = DSIND(ALP)

```

```

C      DEFINE ANTENNA BORESIGHT AK

```

```

C      AK(1) = -ACALP * ASBET
C      AK(2) = ASALP
C      AK(3) = ACALP * ACBET

```

```

C   DEFINE ANTENNA RANGE TRANSMIT VERTICAL POL VT
C   VT(1) = ASALP * ASBET
C   VT(2) = ACALP
C   VT(3) = -ASALP * ACBET
C   DEFINE ANTENNA RANGE TRANSMIT HORIZONTAL POL HT
C   HT(1) = ACBET
C   HT(2) = 0.D0
C   HT(3) = ASBET
C   these are the az over el matrix values, WLL 11/00 corrected
      AK(1) = - ASBET
      AK(2) = ASALP * ACBET
      AK(3) = ACALP * ACBET
      VT(1) = 0.D0
      VT(2) = ACALP
      VT(3) = -ASALP
      HT(1) = ACBET
      HT(2) = ASALP * ASBET
      HT(3) = ACALP * ASBET

C*****
C*****
C   GET COMPLEX ANTENNA GAIN FUNCTIONS
      GVTPA_A=vcomag(ifreq,j,i)
      GHTPA_A=vcrossmag(ifreq,j,i)
      GVTQA_A=hcrossmag(ifreq,j,i)
      GHTQA_A=hcomag(ifreq,j,i)

      PVTPA_P=vcophase(ifreq,j,i)
      PHTPA_P=vcrossphase(ifreq,j,i)
      PVTQA_P=hcrossphase(ifreq,j,i)
      PHTQA_P=hcophase(ifreq,j,i)

      CPVTPA = DCMPLX(DCOSD(PVTPA_P),DSIND(PVTPA_P))
      CPHTPA = DCMPLX(DCOSD(PHTPA_P),DSIND(PHTPA_P))
      CPVTQA = DCMPLX(DCOSD(PVTQA_P),DSIND(PVTQA_P))
      CPHTQA = DCMPLX(DCOSD(PHTQA_P),DSIND(PHTQA_P))

C   FORM COMPLEX EFFECTIVE HEIGHT VECTORS
      DO L=1,3
      CEVPOLT(L) = GVTPA_A*CPVTPA * VT(L) + GHTPA_A*CPHTPA * HT(L)
      CEHPOLT(L) = GVTQA_A*CPVTQA * VT(L) + GHTQA_A*CPHTQA * HT(L)
      ENDDO

      IF(IANS_PRO.EQ.1)THEN
C   GET LOCAL GEODETIC VE,HE VECTORS FOR VPOL
      CALL GEODETIC(AK,VE,HE)

C   ROTATE COMPLEX EFFECTIVE HEIGHT VECTORS
C   note: for no rotation, ACPOLROT =1 and ASPOLROT =0
      DO L=1,3
      CEVPOL(L) = ACPOLROT * CEVPOLT(L) + ASPOLROT * CEHPOLT(L)
      CEHPOL(L) = -ASPOLROT * CEVPOLT(L) + ACPOLROT * CEHPOLT(L)
      ENDDO

      ELSE
C   WE USE ANTENNA RANGE POLARIZATION BASIS

      DO L=1,3
      VE(L) = VT(L)
      HE(L) = HT(L)
      CEVPOL(L) = CEVPOLT(L)
      CEHPOL(L) = CEHPOLT(L)
      ENDDO

      ENDIF

C   FORM COEFS BIGA,BIGB,BIGC,BIGD
      CBIGA = DCMPLX(0.D0,0.D0)
      CBIGB = DCMPLX(0.D0,0.D0)
      CBIGC = DCMPLX(0.D0,0.D0)

```

```

CBIGD = DCMPLX(0.D0,0.D0)
DO L=1,3
CBIGA = CBIGA + CEVPOL(L) * VE(L)
CBIGB = CBIGB + CEVPOL(L) * HE(L)
CBIGC = CBIGC + CEHPOL(L) * VE(L)
CBIGD = CBIGD + CEHPOL(L) * HE(L)
ENDDO

C  NORMALIZE CBIGA,B,C,D
CBIGA = CBIGA/ANORM_V
CBIGB = CBIGB/ANORM_V
CBIGC = CBIGC/ANORM_H
CBIGD = CBIGD/ANORM_H

C  COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO VA
GVEVA = 0.5D0 * CDABS(CBIGA + CBIGC)**2
GHEVA = 0.5D0 * CDABS(CBIGB + CBIGD)**2
GUEVA = 0.5D0 * DREAL((CBIGA+CBIGC)*DCONJG(CBIGB+CBIGD))
G4EVA = -0.5D0 * DIMAG((CBIGA+CBIGC)*DCONJG(CBIGB+CBIGD))

C  COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO HA
GVEHA = 0.5D0 * CDABS(CBIGA - CBIGC)**2
GHEHA = 0.5D0 * CDABS(CBIGB - CBIGD)**2
GUEHA = 0.5D0 * DREAL((CBIGA-CBIGC)*DCONJG(CBIGB-CBIGD))
G4EHA = -0.5D0 * DIMAG((CBIGA-CBIGC)*DCONJG(CBIGB-CBIGD))

C  COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO UA
GVEUA = CDABS(CBIGA)**2 - CDABS(CBIGC)**2
GHEUA = CDABS(CBIGB)**2 - CDABS(CBIGD)**2
GUEUA = DREAL(CBIGA*DCONJG(CBIGB) - CBIGC*DCONJG(CBIGD))
G4EUA = DIMAG(CBIGC*DCONJG(CBIGD) - CBIGA*DCONJG(CBIGB))

C  COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO 4A
GVE4A = -2.D0 * DIMAG( CBIGA * DCONJG(CBIGC) )
GHE4A = -2.D0 * DIMAG( CBIGB * DCONJG(CBIGD) )
GUE4A = -DIMAG(CBIGA*DCONJG(CBIGD) - CBIGC*DCONJG(CBIGB) )
G4E4A = -DREAL(CBIGA*DCONJG(CBIGD) - CBIGC*DCONJG(CBIGB) )

```

```

C*****
C  now sum up all the terms to get the Stokes matrix:
C  separate out the positive and negative terms as well
C  note, ACALP was changed to ACBET for az over el   WLL 12/00
C*****

```

```

ACOU_PVEA = ACOU_PVEA + GVEVA *ACBET
ACOU_PHEA = ACOU_PHEA + GHEVA *ACBET
ACOU_PUEA = ACOU_PUEA + GUEVA *ACBET
ACOU_P4EA = ACOU_P4EA + G4EVA *ACBET

```

```

ACOU_PVEHA = ACOU_PVEHA + GVEHA *ACBET
ACOU_PHEHA = ACOU_PHEHA + GHEHA *ACBET
ACOU_PUEHA = ACOU_PUEHA + GUEHA *ACBET
ACOU_P4EHA = ACOU_P4EHA + G4EHA *ACBET

```

```

ACOU_PVEUA = ACOU_PVEUA + GVEUA *ACBET
ACOU_PHEUA = ACOU_PHEUA + GHEUA *ACBET
IF(ASBET.GE.0.D0)THEN
ACOU_PVEUA_RT = ACOU_PVEUA_RT + GVEUA *ACBET
ACOU_PHEUA_RT = ACOU_PHEUA_RT + GHEUA *ACBET
ELSE
ACOU_PVEUA_LT = ACOU_PVEUA_LT + GVEUA *ACBET
ACOU_PHEUA_LT = ACOU_PHEUA_LT + GHEUA *ACBET
ENDIF
ACOU_PUEUA = ACOU_PUEUA + GUEUA *ACBET
ACOU_P4EUA = ACOU_P4EUA + G4EUA *ACBET

```

```

ACOU_PVE4A = ACOU_PVE4A + GVE4A *ACBET
ACOU_PHE4A = ACOU_PHE4A + GHE4A *ACBET
IF(ASBET.GE.0.D0)THEN
ACOU_PVE4A_RT = ACOU_PVE4A_RT + GVE4A *ACBET

```

```

ACOU_P_HE4A_RT = ACOU_P_HE4A_RT + GHE4A *ACBET
ELSE
ACOU_P_VE4A_LT = ACOU_P_VE4A_LT + GVE4A *ACBET
ACOU_P_HE4A_LT = ACOU_P_HE4A_LT + GHE4A *ACBET
ENDIF
ACOU_P_UE4A = ACOU_P_UE4A + GUE4A *ACBET
ACOU_P_4E4A = ACOU_P_4E4A + G4E4A *ACBET

ENDDO
ENDDO

```

```

C*****
C   a end of main do loop that finds the Stokes matrix

```

Rastercpnew.for (excerpts)

```

CALL RAST(AK,BVTP_ALP,BVTP_BET,VT,HT)

DO L=1,3
AK0VPOL(L) = AK(L)
CAV0VPOL(L) = (VT(L) + CI * HT(L))/ROOT2
CAH0VPOL(L) = (VT(L) - CI * HT(L))/ROOT2
ENDDO

.....

CALL RAST(AK,BHTQ_ALP,BHTQ_BET,VT,HT)

DO L=1,3
AK0HPOL(L) = AK(L)
CAV0HPOL(L) = (VT(L) + CI * HT(L))/ROOT2
CAH0HPOL(L) = (VT(L) - CI * HT(L))/ROOT2
ENDDO
C   save centroid information so can print out average later
wBHTQ_BET(ifreqloop)=BHTQ_BET
wBHTQ_ALP(ifreqloop)=BHTQ_ALP
wBVTP_BET(ifreqloop)=BVTP_BET
wBVTP_ALP(ifreqloop)=BVTP_ALP

WRITE(50,3341)freqprint,SNGL(-BHTQ_BET),SNGL(-BHTQ_ALP),
1 SNGL(-BHTQ_BET+azencoder),SNGL(-BHTQ_ALP+elencoder)

WRITE(36,3341)freqprint,SNGL(-BHTQ_BET),SNGL(-BHTQ_ALP),
1 SNGL(-BHTQ_BET+azencoder),SNGL(-BHTQ_ALP+elencoder)
3341   FORMAT(7HLC-POL,2X,F6.3,4(F13.4))

C*****
C   COMPLETED CENTROID COMPUTATIONS
C*****
C   WRITE(*,*) ' TO PROJECT ANTENNA BEAMS ONTO EARTH, ENTER 1:'
C   WRITE(*,*) '     ANTENNA BEAMS ONTO ANT RANGE,  2:'
C   WRITE(*,*) ' ENTER SELECTION:'
C   READ(55,*) IANS_PRO
IANS_PRO=1
IF(IANS_PRO.EQ.1)THEN
WRITE(50,*) ' WE PROJECT ANTENNA BEAMS ONTO EARTH SURFACE:'

CALL GEODETIC(AK0VPOL,VE,HE)
DO L=1,3
CPE(L)=( VE(L) + CI * HE(L) )/ROOT2
CAME(L)=( VE(L) - CI * HE(L) )/ROOT2
ENDDO

```

```

CACPOLROTV = DCMPLX(0.D0,0.D0)
CACPOLROTH = DCMPLX(0.D0,0.D0)
DO L=1,3
CACPOLROTV = CACPOLROTV + CAV0VPOL(L)*CPE(L)
CACPOLROTH = CACPOLROTH + CAV0VPOL(L)*CAME(L)
ENDDO

WRITE(66,*) ' RCP * RCPE:'
REPRO = DREAL(CACPOLROTV)
AIPRO = DIMAG(CACPOLROTV)

WRITE(66,*) ' RCP * LCPE:'
REPRO = DREAL(CACPOLROTH)
AIPRO = DIMAG(CACPOLROTH)

CALL GEODETIC(AK0HPOL,VE,HE)
DO L=1,3
CPE(L) =( VE(L) + CI * HE(L) )/ROOT2
CAME(L)=( VE(L) - CI * HE(L) )/ROOT2
ENDDO

CACPOLROTV = DCMPLX(0.D0,0.D0)
CACPOLROTH = DCMPLX(0.D0,0.D0)
DO L=1,3
CACPOLROTV = CACPOLROTV + CAV0HPOL(L)*CPE(L)
CACPOLROTH = CACPOLROTH + CAV0HPOL(L)*CAME(L)
ENDDO

WRITE(66,*) ' LCP * RCPE:'
REPRO = DREAL(CACPOLROTV)
AIPRO = DIMAG(CACPOLROTV)
WRITE(66,*) SNGL(REPRO),SNGL(AIPRO)
WRITE(66,*) ' '

WRITE(66,*) ' LCP * LCPE:'
REPRO = DREAL(CACPOLROTH)
AIPRO = DIMAG(CACPOLROTH)
WRITE(66,*) SNGL(REPRO),SNGL(AIPRO)

ENDIF

C
C*****
C NEXT WE FIND THE NORMALIZATION FACTORS FOR THE ANTENNA
C BEAM ENERGIES

ANORM_V = 0.D0
ANORM_H = 0.D0

DO J=1,numaz
C BET = AZRX(J)
C ACBET = DCOSD(BET)
C ASBET = DSIND(BET)

DO I=1,numel
C ALP = ELR(I)
ALP = elevangle(1,j,i)
ACALP = DCOSD(ALP)
ASALP = DSIND(ALP)
BET = aziangle(1,j,i)
ACBET = DCOSD(BET)
ASBET = DSIND(BET)

C DEFINE ANTENNA BORESIGHT AK
C AK(1) = -ACALP * ASBET
C AK(2) = ASALP
C AK(3) = ACALP * ACBET
C DEFINE ANTENNA RANGE TRANSMIT VERTICAL POL VT
C VT(1) = ASALP * ASBET
C VT(2) = ACALP
C VT(3) = -ASALP * ACBET

```



```

C   DEFINE ANTENNA RANGE TRANSMIT HORIZONTAL POL HT
C   HT(1) = ACBET
C   HT(2) = 0.D0
C   HT(3) = ASBET
C   these are the az over el matrix values, WLL 11/00 corrected
      AK(1) = - ASBET
      AK(2) = ASALP * ACBET
      AK(3) = ACALP * ACBET
      VT(1) = 0.D0
      VT(2) = ACALP
      VT(3) = -ASALP
      HT(1) = ACBET
      HT(2) = ASALP * ASBET
      HT(3) = ACALP * ASBET

C*****
C   GET COMPLEX ANTENNA GAIN FUNCTIONS

      GVTPA_A=vcomag(ifreq,j,i)
      GHTPA_A=vcrossmag(ifreq,j,i)
      GVTQA_A=hcrossmag(ifreq,j,i)
      GHTQA_A=hcomag(ifreq,j,i)

      PVTPA_P=vcphase(ifreq,j,i)
      PHTPA_P=vcrossphase(ifreq,j,i)
      PVTQA_P=hcrossphase(ifreq,j,i)
      PHTQA_P=hcophase(ifreq,j,i)

      CPVTPA = DCMPLX(DCOSD(PVTPA_P),DSIND(PVTPA_P))
      CPHTPA = DCMPLX(DCOSD(PHTPA_P),DSIND(PHTPA_P))
      CPVTQA = DCMPLX(DCOSD(PVTQA_P),DSIND(PVTQA_P))
      CPHTQA = DCMPLX(DCOSD(PHTQA_P),DSIND(PHTQA_P))

C   TEMPORARY
      CPVTPA = CPVTPA * CIM
      CPHTPA = CPHTPA * CIM

C   FORM COMPLEX EFFECTIVE HEIGHT VECTORS
      DO L=1,3
      CEVPOL(L) = GVTPA_A*CPVTPA * VT(L) + GHTPA_A*CPHTPA * HT(L)
      CEHPOL(L) = GVTQA_A*CPVTQA * VT(L) + GHTQA_A*CPHTQA * HT(L)
      VE(L) = VT(L)
      HE(L) = HT(L)
      ENDDO

C   FORM COEFS BIGA,BIGB,BIGC,BIGD
      CBIGA = DCMPLX(0.D0,0.D0)
      CBIGB = DCMPLX(0.D0,0.D0)
      CBIGC = DCMPLX(0.D0,0.D0)
      CBIGD = DCMPLX(0.D0,0.D0)
      DO L=1,3
      CBIGA = CBIGA + CEVPOL(L) * VE(L)
      CBIGB = CBIGB + CEVPOL(L) * HE(L)
      CBIGC = CBIGC + CEHPOL(L) * VE(L)
      CBIGD = CBIGD + CEHPOL(L) * HE(L)
      ENDDO

C   DO SUMS
C   these are for elevation over azimuth:
      ANORM_V=ANORM_V+(CDABS(CBIGA)**2+CDABS(CBIGB)**2)*ACALP
      ANORM_H=ANORM_H+(CDABS(CBIGC)**2+CDABS(CBIGD)**2)*ACALP
C   this is for azimuth over elevation:
      ANORM_V=ANORM_V+(CDABS(CBIGA)**2+CDABS(CBIGB)**2)*ACBET
      ANORM_H=ANORM_H+(CDABS(CBIGC)**2+CDABS(CBIGD)**2)*ACBET
      ANORM_Vwl=ANORM_Vwl+(CDABS(GVTPA_A*CPVTPA)**2
1 +CDABS(GHTPA_A*CPHTPA)**2)
      ANORM_Hwl=ANORM_Hwl+(CDABS(GVTQA_A*CPVTQA)**2
1 +CDABS(GHTQA_A*CPHTQA)**2)

      ENDDO

```

```

ENDDO

ANORM_V = DSQRT(ANORM_V)
ANORM_H = DSQRT(ANORM_H)

WRITE(50,*) 'NORMALIZED ENERGIES:'
WRITE(50,*) ' RCP,LCP:',SNGL(ANORM_V),SNGL(ANORM_H)
WRITE(31,*) ifreqloop,SNGL(ANORM_V),SNGL(ANORM_H),' RCP LCP:'

*****
C   COMPUTE COUPLING MATRIX OF STOKES PARAMETERS
*****
C   INITIALIZE SUMS TO ZERO
ACOUN_VEVA = 0.D0
ACOUN_HEVA = 0.D0
ACOUN_UEVA = 0.D0
ACOUN_4EVA = 0.D0

ACOUN_VEHA = 0.D0
ACOUN_HEHA = 0.D0
ACOUN_UEHA = 0.D0
ACOUN_4EHA = 0.D0

ACOUN_VEUA_RT = 0.D0
ACOUN_HEUA_RT = 0.D0
ACOUN_VEUA_LT = 0.D0
ACOUN_HEUA_LT = 0.D0
ACOUN_VEUA = 0.D0
ACOUN_HEUA = 0.D0
ACOUN_UEUA = 0.D0
ACOUN_4EUA = 0.D0

ACOUN_VE4A_RT = 0.D0
ACOUN_HE4A_RT = 0.D0
ACOUN_VE4A_LT = 0.D0
ACOUN_HE4A_LT = 0.D0
ACOUN_VE4A = 0.D0
ACOUN_HE4A = 0.D0
ACOUN_UE4A = 0.D0
ACOUN_4E4A = 0.D0

DO J=1,numaz
C   BET = AZRX(J)
C   ACBET = DCOSD(BET)
C   ASBET = DSIND(BET)

DO I=1,numel
C   ALP = ELR(I)
ALP = elevangle(1,j,i)
ACALP = DCOSD(ALP)
ASALP = DSIND(ALP)
BET = aziangle(1,j,i)
ACBET = DCOSD(BET)
ASBET = DSIND(BET)

C   DEFINE ANTENNA BORESIGHT AK
C   AK(1) = -ACALP * ASBET
C   AK(2) = ASALP
C   AK(3) = ACALP * ACBET
C   DEFINE ANTENNA RANGE TRANSMIT VERTICAL POL VT
C   VT(1) = ASALP * ASBET
C   VT(2) = ACALP
C   VT(3) = -ASALP * ACBET
C   DEFINE ANTENNA RANGE TRANSMIT HORIZONTAL POL HT
C   HT(1) = ACBET
C   HT(2) = 0.D0
C   HT(3) = ASBET
C   these are the az over el matrix values, WLL 11/00 corrected
C   AK(1) = - ASBET

```

```

AK(2) = ASALP * ACBET
AK(3) = ACALP * ACBET
VT(1) = 0.D0
VT(2) = ACALP
VT(3) = -ASALP
HT(1) = ACBET
HT(2) = ASALP * ASBET
HT(3) = ACALP * ASBET

```

C*****

C GET COMPLEX ANTENNA GAIN FUNCTIONS

```

GVTPA_A=vcomag(ifreq,j,i)
GHTPA_A=vcrossmag(ifreq,j,i)
GVTQA_A=hcrossmag(ifreq,j,i)
GHTQA_A=hcomag(ifreq,j,i)

```

```

PVTPA_P=vcophase(ifreq,j,i)
PHTPA_P=vcrossphase(ifreq,j,i)
PVTQA_P=hcrossphase(ifreq,j,i)
PHTQA_P=hcophase(ifreq,j,i)

```

```

CPVTPA = DCMPLX(DCOSD(PVTPA_P),DSIND(PVTPA_P))
CPHTPA = DCMPLX(DCOSD(PHTPA_P),DSIND(PHTPA_P))
CPVTQA = DCMPLX(DCOSD(PVTQA_P),DSIND(PVTQA_P))
CPHTQA = DCMPLX(DCOSD(PHTQA_P),DSIND(PHTQA_P))

```

C TEMPORARY

```

CPVTPA = CPVTPA * CIM
CPHTPA = CPHTPA * CIM

```

C FORM COMPLEX EFFECTIVE HEIGHT VECTORS

```

DO L=1,3
CEVPOL(L) = GVTPA_A*CPVTPA * VT(L) + GHTPA_A*CPHTPA * HT(L)
CEHPOL(L) = GVTQA_A*CPVTQA * VT(L) + GHTQA_A*CPHTQA * HT(L)
ENDDO

```

IF(IANS_PRO.EQ.1)THEN

C GET LOCAL GEODETIC VE,HE VECTORS FOR VPOL
CALL GEODETIC(AK,VE,HE)

ELSE

C USE ANTENNA RANGE RCP,LCP

```

DO L=1,3
VE(L) = VT(L)
HE(L) = HT(L)
ENDDO

```

ENDIF

C FORM COEFS CBIGA,BIGB,BIGC,BIGD

```

CBIGA = DCMPLX(0.D0,0.D0)
CBIGB = DCMPLX(0.D0,0.D0)
CBIGC = DCMPLX(0.D0,0.D0)
CBIGD = DCMPLX(0.D0,0.D0)
DO L=1,3
CBIGA = CBIGA + CEVPOL(L) * VE(L)
CBIGB = CBIGB + CEVPOL(L) * HE(L)
CBIGC = CBIGC + CEHPOL(L) * VE(L)
CBIGD = CBIGD + CEHPOL(L) * HE(L)
ENDDO

```

C NORMALIZE CBIGA,B,C,D

```

CBIGA = CBIGA/ANORM_V
CBIGB = CBIGB/ANORM_V
CBIGC = CBIGC/ANORM_H
CBIGD = CBIGD/ANORM_H

```

C COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO VA

```

GVEVA = 0.5D0 * CDABS(CBIGA + CBIGC)**2

```

```

      GHEVA = 0.5D0 * CDABS(CBIGB + CBIGD)**2
      GUEVA = 0.5D0 * DREAL((CBIGA+CBIGC)*DCONJG(CBIGB+CBIGD))
      G4EVA = -0.5D0 * DIMAG((CBIGA+CBIGC)*DCONJG(CBIGB+CBIGD))

C      COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO HA
      GVEHA = 0.5D0 * CDABS(CBIGA - CBIGC)**2
      GHEHA = 0.5D0 * CDABS(CBIGB - CBIGD)**2
      GUEHA = 0.5D0 * DREAL((CBIGA-CBIGC)*DCONJG(CBIGB-CBIGD))
      G4EHA = -0.5D0 * DIMAG((CBIGA-CBIGC)*DCONJG(CBIGB-CBIGD))

C      COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO UA
      GVEUA = -2.D0* DIMAG( CBIGA * DCONJG(CBIGC) )
      GHEUA = -2.D0* DIMAG( CBIGB * DCONJG(CBIGD) )
      GUEUA = -DIMAG(CBIGA*DCONJG(CBIGD) - CBIGC*DCONJG(CBIGB))
      G4EUA = -DREAL(CBIGA*DCONJG(CBIGD) - CBIGC*DCONJG(CBIGB) )

C      COMPUTE COUPLING BETWEEN VE,HE,UE,4E TO 4A
      GVE4A = CDABS(CBIGA)**2 - CDABS(CBIGC)**2
      GHE4A = CDABS(CBIGB)**2 - CDABS(CBIGD)**2
      GUE4A = DREAL(CBIGA*DCONJG(CBIGB) - CBIGC*DCONJG(CBIGD))
      G4E4A = DIMAG(CBIGA*DCONJG(CBIGB) - CBIGC*DCONJG(CBIGD))

```

```

C*****
C      now sum up all the terms to get the Stokes matrix:
C      separate out the positive and negative terms as well
C      note, ACALP was changed to ACBET for az over el   WLL 12/00
C*****

```

```

      ACOUP_VEVA = ACOUP_VEVA + GVEVA * ACBET
      ACOUP_HEVA = ACOUP_HEVA + GHEVA * ACBET
      ACOUP_UEVA = ACOUP_UEVA + GUEVA * ACBET
      ACOUP_4EVA = ACOUP_4EVA + G4EVA * ACBET

```

```

      ACOUP_VEHA = ACOUP_VEHA + GVEHA * ACBET
      ACOUP_HEHA = ACOUP_HEHA + GHEHA * ACBET
      ACOUP_UEHA = ACOUP_UEHA + GUEHA * ACBET
      ACOUP_4EHA = ACOUP_4EHA + G4EHA * ACBET

```

```

      ACOUP_VEUA = ACOUP_VEUA + GVEUA * ACBET
      ACOUP_HEUA = ACOUP_HEUA + GHEUA * ACBET
      IF(ASBET.GE.0.D0)THEN
        ACOUP_VEUA_RT = ACOUP_VEUA_RT + GVEUA * ACBET
        ACOUP_HEUA_RT = ACOUP_HEUA_RT + GHEUA * ACBET
      ELSE
        ACOUP_VEUA_LT = ACOUP_VEUA_LT + GVEUA * ACBET
        ACOUP_HEUA_LT = ACOUP_HEUA_LT + GHEUA * ACBET
      ENDIF
      ACOUP_UEUA = ACOUP_UEUA + GUEUA * ACBET
      ACOUP_4EUA = ACOUP_4EUA + G4EUA * ACBET

```

```

      ACOUP_VE4A = ACOUP_VE4A + GVE4A * ACBET
      ACOUP_HE4A = ACOUP_HE4A + GHE4A * ACBET
      IF(ASBET.GE.0.D0)THEN
        ACOUP_VE4A_RT = ACOUP_VE4A_RT + GVE4A * ACBET
        ACOUP_HE4A_RT = ACOUP_HE4A_RT + GHE4A * ACBET
      ELSE
        ACOUP_VE4A_LT = ACOUP_VE4A_LT + GVE4A * ACBET
        ACOUP_HE4A_LT = ACOUP_HE4A_LT + GHE4A * ACBET
      ENDIF
      ACOUP_UE4A = ACOUP_UE4A + GUE4A * ACBET
      ACOUP_4E4A = ACOUP_4E4A + G4E4A * ACBET

```

```

      ENDDO
      ENDDO

```

```

C*****
C      a end of main do loop that finds the Stokes matrix
C*****
C
C

```